TRIUMF	4004 Wesbrook Mall, Vancouver, B.C., Canada V6T 2A3		
Design Note	F.W. Jones	June 1990	TRI-DN-90-17
User's Guide to ACCSIM			

User's Guide to ACCSIM

F.W. Jones

TRIUMF
4004 Wesbrook Mall
Vancouver, B.C.
Canada V6T 2A3

Version 3.0
June 1990

ACCSIM was developed by

F.W. Jones, TRIUMF
G.H. Mackenzie, TRIUMF
H.O. Schönauer, CERN [Transverse space charge package]

Acknowledgments

This program has benefited from the ideas and contributions of a number of people. Deepak Raparia helped in getting the early version operational and added the Landau distribution for energy loss. Maria Masullo collaborated on improvements to the code and facilitated the use of the CERN computing environment. Shane Koscielniak's expertise on longitudinal space charge was essential in implementing that option, and his work on scattering cross sections was also helpful. Rick Baartman has provided advice on some important aspects of the mathematical models. He was instrumental in generalizing the transport algorithm and more recently has resolved problems in the space-charge tune shift implementation. Roger Servranckx offered guidance on tracking techniques and his program DIMAD has been a primary resource. Work done by Arch Thiessen and others at LAMPF, including the program ARCHSIM, was influential in the development of the present program. This project was greatly helped by the availability of expanded computing hardware and by a variety of software resources at TRIUMF, and Corrie Kost's work on this behalf is much appreciated.

Additional Documentation

In addition to this guide the following references can be consulted for descriptions of the program design:

- F.W. Jones, G.H. Mackenzie, and H. Schönauer, "ACCSIM—A program to simulate the accumulation of intense proton beams," *Proc. 14th International Conference on High Energy Accelerators*, in *Particle Accelerators* **31:**199 (1990).
- H. Schönauer, "Addition of Transverse Space Charge to ACCSIM Code," Triumf Design Note TRI-DN-89-K50, 1989.
- R. Baartman and F.W. Jones, "Transport Equations for ACCSIM," Triumf Design Note TRI-DN-89-K62, 1989.

Contents

I Program Overview 1

1 Introduction 1

2 Operation 1

3 Fundamentals 2

4 Machine Model 3

5 Functional Summary 4

6 Input Files 5
- 6.1 Main input file 6
- 6.2 Lattice File 6
  - 6.2.1 Format of DIMAD run 7
- 6.3 Distribution input file 9
- 6.4 Command input stream 9
- 6.5 Plotting parameters file 9
- 6.6 Save file input 9

7 Output Files 10
- 7.1 Echo output 10
- 7.2 Main output 10
- 7.3 Plot file 10
- 7.4 Injection coordinates file 11
- 7.5 Tabular record file 11
- 7.6 Save file 11
- 7.7 Auxiliary Output Files 11

8 Preparing a Run 11

9 Injected Beam 13
- 9.1 Distributions 13
- 9.2 Injection coordinates 15
- 9.3 Dispersion 15
- 9.4 Painting programs 16

10 Macroparticle Injection 17

11 Injection Foil **18**
11.1 Coulomb scattering . . . 18
11.2 Multiple scattering . . . 19
11.3 Plural scattering . . . 19
11.4 Single scattering . . . 20
11.5 Scattering cross sections . . . 21
11.6 Energy loss . . . 22
11.7 Nuclear scattering . . . 23
11.8 Foil histograms . . . 23

12 Internal Target **24**

13 Inter-node Transport **24**

14 Node Properties **25**
14.1 RF cavity . . . 26
14.1.1 Rf harmonics . . . 27
14.2 Longitudinal space charge . . . 27
14.3 Thin lens . . . 30
14.4 Aperture or collimation slit . . . 31
14.5 Plot or track output . . . 32

15 Transverse Space Charge **32**
15.1 Tune-shift rotation . . . 33
15.2 Tune-shift transformations . . . 33

16 Chromaticity **35**

17 Tabular Output **35**

18 Graphics **36**

19 Pre- and Post-processing **37**

20 Program Design **39**
20.1 Computational core . . . 40
20.2 Principal routines . . . 42

21 Performance Issues **43**
21.1 Run times . . . 43
21.2 Benchmarks . . . 44

22 Modifying ACCSIM **44**

II Reference Guide **47**

23 Main Input File **47**

24 BEAM Parameter List **49**
24.1 Transverse distributions 49
24.2 Longitudinal distribution 50
24.3 Dispersion matching 50
24.4 Injected beam center 50
24.5 Distributions read from file 51
24.6 Closed orbit bump 51
24.7 Painting 51

25 NODES Parameter List **53**

26 MAIN Parameter List **55**
26.1 Vacuum chamber 55
26.2 RF System 55
26.3 Foil or target 56
26.4 Macroparticle injection 57
26.5 Line density 57
26.6 Space charge 57
26.7 Chromaticity 58
26.8 Acceptance 58
26.9 Histograms and plotting 58
26.10Emittance fractions 58
26.11Random number generator 58

27 OPTIONS Parameter List **59**
27.1 Option switches 59
27.2 Transverse space charge diagnostics 59

28 Interactive Commands **61**
28.1 Tracking and analysis commands 61
28.2 Graphics commands 63

29 Plotting Parameters **65**
29.1 GPLOT parameters 66

30 I/O Streams **69**
30.1 File naming convention 70

31 Output Formats **71**

References **73**

Appendix: Example Run **77**

List of Figures

1 Local coordinate system 3
2 Example machine model 4
3 Example painting function 17
4 Comparison of scattering models 22
5 Transport between nodes. 24
6 OPDATA/EDGR example. 39

Part I
Program Overview

1 Introduction

ACCSIM is a multiparticle tracking and simulation code that is primarily used to study multiturn charge-exchange injection into an accumulator ring. It can also be used in various other applications which have included internal targets, rf capture and other longitudinal studies, transverse painting by linear coupling, collimation and loss processes, and estimation of diagnostic signals.

This program does not offer the accuracy or analysis capabilities of codes like DIMAD or MAD, but is useful in situations where the tracking of ensembles of a few thousand particles needs to be integrated efficiently with the modelling of effects like foil scattering or space charge. In scope it applies to proton synchrotrons or storage rings, where the basic magnet lattice is described in the standard MAD input language and additional nonlinear or skew elements are treated as thin lenses. Although rf cavities are simulated, the treatment of beam acceleration is not yet included: it is assumed that the ring is operating at a fixed reference energy and that the synchronous phase is zero.

A combination of matrix/thin-lens tracking and Monte Carlo simulation is used to follow the ensemble of particles as it evolves in the machine. Various optional functions are included to simulate foil or target scattering and energy loss, longitudinal and transverse space-charge effects, chromaticity, and collimators or apertures at which losses can be monitored. A number of form factors describing beam distribution and quality are computed at regular intervals. Multiturn injection is modelled by adding new particles to the ensemble at intervals during the run. In this case injection "painting" schemes can be included, whereby injection parameters are specified as time-varying functions. Injected or circulating beams can be chosen from a range of distributions, and graphics facilities are provided to produce scatterplots of the beam at any time during the run.

The auxiliary analysis capabilities of the program are limited, but virtually all of the program's internal data can be output to files for external processing. This has allowed the code to be kept to modest proportions and takes advantage of the powerful stand-alone analysis/graphics packages that are now available.

2 Operation

ACCSIM runs on VAX/VMS systems and requires no special installation procedures. A typical run consists of preparing a set of input files, assigning files to the input and output streams, and running the program in either batch or interactive mode. During the run, program action is controlled by the input parameters and by commands typed by the user at the terminal or placed in the batch file. Text and graphics output is directed to the screen (in interactive mode) and to files for later printing.

The command interface provides considerable flexibility in the progress of a run. In interactive mode one can operate the simulated machine for any desired number of turns and obtain coordinate readouts, scatterplots and statistics at any time. This makes experimentation easier and allows one to check the initial behaviour of a long job before proceeding to the full batch run.

Of note are the **`SAVE`** and **`RESTORE`** functions which allow the entire context of the simulation to be written to a file and then reinstated when desired. For example, the final results of a batch run can be saved and then restored in an interactive run to produce additional statistics or scatterplots or to continue machine operation for more turns. Also, the saving of intermediate results allows re-start of a run if there is a hardware failure.

3 Fundamentals

The machine geometry is described in the standard coordinate system used by programs such as TRANSPORT and DIMAD, with local coordinates referred to a *reference orbit* which is the orbit followed by a particle with the central design momentum (reference momentum) in the absence of magnet errors or fringe fields. It is assumed that the rf frequency is locked to the nominal revolution period of the reference orbit, so that a particle with the reference momentum is rf-synchronous and we can refer to the synchronous momentum, p_s, and synchronous energy, E_s, as identical with the reference values.

For particle tracking ACCSIM uses the following basic set of variables:

x	Horizontal distance from the reference orbit (mm)
x'	Horizontal angle with respect to the reference orbit (mr)
y	Vertical distance from the reference orbit (mm)
y'	Vertical angle with respect to the reference orbit (mr)
ϕ	Rf phase (deg) – phase of the rf voltage generator when the particle crosses an acceleration gap.
ΔE	Energy difference $E - E_s$ (MeV) with respect to a synchronous particle

As shown in Figure 1, the orientation of the coordinate system is such that a particle with positive x lies to the outside of the reference orbit (with respect to the machine center) and a particle with positive y lies above the reference orbit.

Note that $\phi, \Delta E$ are used instead of the usual transport variables ℓ, δ. This choice of coordinates facilitates the inclusion of rf cavities in the tracking process. The transport coordinate ℓ refers to path length difference and does not directly give the time difference needed to determine rf phase (see Section 13 and Reference [3] for further discussion). The coordinate $\delta = \Delta p/p_s$ can be obtained from ΔE and E_s so that only ΔE needs to be stored by the program. To first order, the relation $dp/p = \beta^{-2}dE/E$ can be used, whereas an exact conversion can be used in higher-order calculations.

For convenience, the program handles transverse distances and angles in the typical working units of millimeters and milliradians. For compatibility with these units the

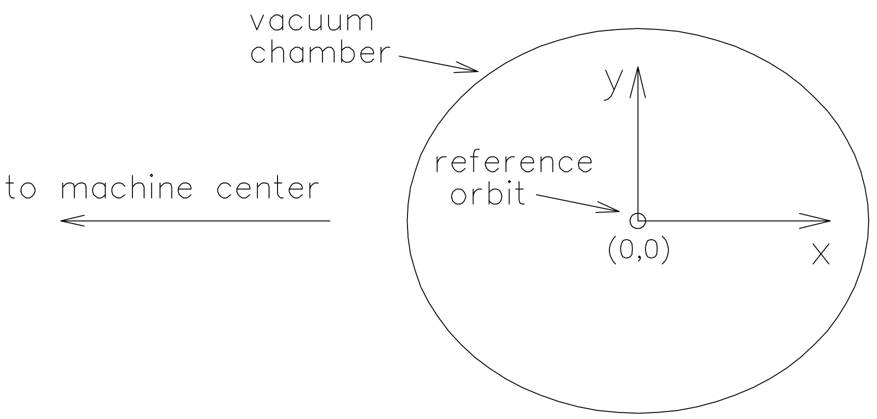

Figure 1: Local coordinate system (beam moving out of page).

transport coordinate $\Delta p/p$ is specified in units of 10^{-3} at the appropriate places in the code.

4 Machine Model

The main task of ACCSIM is to follow an ensemble of *macroparticles*, considered to be a representative sample of protons, around the machine for many turns. (In this document, the terms "macroparticle" and "particle" are used more-or-less interchangeably, whereas "proton" or "real particle" means the genuine article.)

The machine being simulated is primarily defined by a magnet lattice that begins and ends at the injection point. The term "injection point" need not always be taken literally: it refers generally to the point at which the calculation of each turn starts and ends. New particles are added to the ensemble at this point. This can be done on a turn-by-turn basis to simulate multiturn injection, or a complete ensemble can be put in before tracking begins.

If required, a stripping foil can be included at the injection point. The effects of scattering and energy loss will be simulated for particles hitting the foil. Alternatively, an internal target can be simulated.

In addition to the injection point, the lattice is punctuated by various other points of interest called *nodes*. Particle coordinates are updated at each node as the ensemble is followed around the ring. The nodes allow additional machine elements or other properties to be included in the machine. Examples of characteristics of a node are:

- An RF cavity is located at the node.
- A longitudinal space-charge calculation is made at the node and energy kicks are applied to the particles.
- A thin lens, up to octupole order, is located at the node.

- An aperture check, simulating a collimator or other constraint, is performed at the node and lost particles are tagged.
- A collimation slit is located at the node. Particles hitting the slit material will undergo multiple scattering and energy loss.
- Particle coordinates at the node are plotted or output to a file.

A given node can have any or all of the properties listed above. At present, up to 100 nodes are allowed. The injection point is always Node 1, and the other nodes are numbered in sequence around the ring.

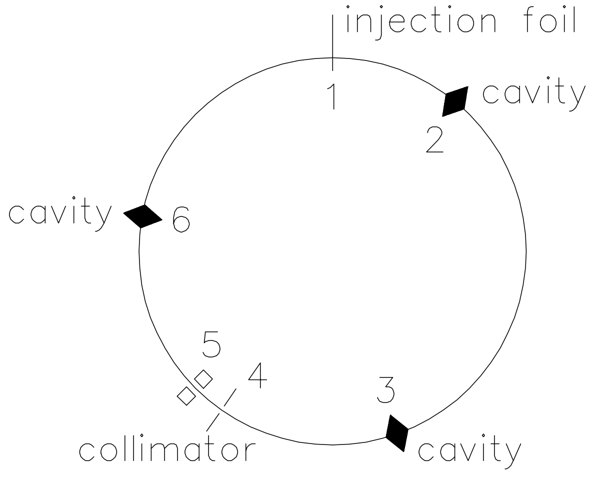

Figure 2: Example machine model with 6 nodes: injection foil (1), three cavities (2,3,6) and a collimation system with a slit (4) and collector (5).

To describe the basic magnet lattice, to which the nodes are added, the program DIMAD[4] is used. This allows the specification of the lattice in the *standard input language* originally used by the MAD program and now adopted by several others. Prior to making ACCSIM runs, a DIMAD input file is prepared for the lattice to be studied, and a DIMAD run is performed to generate the transfer matrices and lattice functions used by ACCSIM. The DIMAD output file for the run is automatically scanned by ACCSIM to extract the needed matrices and parameters.

5 Functional Summary

The various tasks of the ACCSIM program can be summarized as follows:

- Read input files and DIMAD output file for lattice.
- Generate macroparticles populating specified distributions for the injected or circulating beam.

- Follow the ensemble for specified numbers of turns, adding macroparticles at intervals to simulate multiturn injection.
- Simulate injection painting by ramping injection parameters.
- Simulate scattering and energy loss effects in a stripping foil or internal target.
- Treat multipoles in the lattice by thin-lens approximation.
- Calculate longitudinal and transverse space-charge potentials and apply appropriate kicks or betatron phase adjustments, respectively, to particles.
- Simulate multiple scattering and energy loss in collimation slits.
- Tabulate particle statistics such as loss at apertures or foil traversals. Tabulate beam parameters such as emittance and bunching factor.
- Produce scatterplots of the beam at specified intervals.
- Produce auxiliary output files containing particle coordinates, test particle tracking data, emittances, etc., for post-run analysis and plotting.

6 Input Files

An ACCSIM run requires several input files, attached to the following I/O units:

Unit 1: Main input file A description of the injected beam, the node structure of the machine, and machine parameters.

Unit 3: Lattice file A DIMAD output file describing the lattice.

Unit 4: Distribution input A list of particle coordinates for the injected beam distribution, used only if distributions are not generated by ACCSIM.

Unit 5: Command input A series of commands controlling the progress of the run, read from the terminal or from a batch command file.

Unit 8: Plotting parameters A list of parameters controlling the appearance of plots, required only if scatterplots are produced.

Unit 12: Save file input This stream is used to restore a run by reading in a `SAVE` file.

The input file formats are described in detail in Part II of this guide. The following sections give general descriptions.

6.1 Main input file

This file consists of a set of parameter lists describing the following aspects of the run:

1. Injected beam distributions and painting schemes
2. The list of nodes and properties of each node
3. General parameters for the machine including rf, space charge, macroparticle injection, plotting, etc.
4. Special features and diagnostic output

In this file, the lists and parameters are tagged by name and can be input in free format. Details of all the parameters are given in Sections 23 through 27

6.2 Lattice File

This file is the output file from a DIMAD[4] run for the machine under consideration. This run is set up in order to generate the following information relevant to ACCSIM:

- A list of the lattice elements (drift, bend, quad, etc.) in the standard MAD input language. For the purposes of ACCSIM, the lattice must begin and end at the injection point.
- The following lattice functions and machine parameters:
  - Matched functions $\alpha_x, \alpha_y, \beta_x, \beta_y, \gamma_x, \gamma_y$ at the injection point
  - Dispersions η_x, η_x' at the injection point
  - Betatron tunes Q_x, Q_y
  - Chromaticicities C_x, C_y
  - Total length of machine L_T
  - Momentum compaction factor α_p

  Note: ACCSIM does not treat machines with vertical dispersion: the functions η_y and η_y' are assumed to be zero.
- First-order transfer matrices from each node to the next node. The product of all these matrices equals the full-turn transfer matrix at the injection point.
- The distances between nodes, and the values of β_x, η_x, η_x' at each node.

6.2.1 Format of DIMAD run

The ACCSIM program contains a procedure READ_LATTICE which scans the DIMAD output and automatically extracts the information listed above. No modification of the output file is required. However, in order for this scan to work correctly, the DIMAD input file for the run must follow a certain sequence of operations.

In the simplest case, the required operations are:

1. Definitions of elements, machine sections, and full machine.
2. USE command to establish the full machine
3. MATR command to generate the full-turn transfer matrix and calculate the above-mentioned machine parameters
4. MACH command to generate the element list with distances and lattice functions at each element
5. MATR command to generate the inter-node transfer matrices

In many cases, there are adjustments and tuning to be done before the final lattice is analyzed. In this case a comment line "*ACCSIM" must be inserted in the DIMAD input file to signal the start of the final lattice data for ACCSIM. The sequence is as follows:

1. Definitions of elements, machine sections, and full machine.
2. Any adjustment or analysis of sub-sections of the machine
3. USE command to establish the full machine
4. Initial MATR command required if fitting is used
5. Fitting or modification commands to do any tuning or other changes to the lattice
6. *ACCSIM comment flag to start reading by ACCSIM
7. MATR command to generate the full-turn transfer matrix and calculate the required machine parameters
8. MACH command to generate the element list with distances and lattice functions at each element
9. MATR command to generate the inter-node transfer matrices

The following is an example input file for such a DIMAD run:

```
TITLE
Accumulator Lattice, n=3, 05-Feb-90
UTRANSPORT
*** Bending magnets
HB1:SBEND,L=1.005/2,ANGLE=15/2,E1=7.5, E2=0
HB2:SBEND, L=1.005/2, ANGLE=15/2, E1=0, E2=7.5
*** Drift spaces
FD:DRIFT,L=3.5-.00003
FD1:DRIFT,L=0.895
  ...
  ... (element definitions continue)
  ...
*** Build the lines
ID1:LINE=(FDI1,KV1,FDI1,KV2,FDI2,KH1,FDI3,KH2,FDIT)
I1:LINE=(FD,QF1,FD1,QD1,FD4,HB1,HB2,FD24,QF2)
  ...
  ... (section definitions continue)
  ...
*** Full machine
FULL:LINE=(SPIB,SPE1,SPE2,SPIA)
USE,FULL
DIMAT
MATR
2 -1,
*** Adjust tunes
SIMPLE
2 2 2
QD   K1   .001
QF2  K1   .001
 2   0.79
12   0.71
1
QD  K1  1
QDH  K1  1  0,
*** Start reading by ACCSIM
*ACCSIM
MATR
2 -1,
MACH
0,
*** Inter-node matrices
MATR
-11 4
1 34
35 107
108 184
185 239;
STOP
```

In this example there are four nodes: the injection point and three rf cavities. Each inter-node matrix is generated by specifying the numbers of the first and last elements in each machine section: 1–34, 35–107, 108–184, and 185–239. The generated matrix covers the section from the *entrance* of the first element to the *exit* of the last element.

6.3 Distribution input file

ACCSIM contains modules for generating various Monte Carlo distributions for the incoming beam. This is the usual way to generate macroparticles to be injected into the machine. However, an option is provided to read macroparticle coordinates from a file generated by some other program. See Section 24.5 for details on the format of this file.

6.4 Command input stream

Once the primary input file and lattice file are read, and preliminary calculations performed, ACCSIM goes into "command mode", where it reads and executes commands read from the input stream (Unit 5). Typical commands are:

`TURN`	Execute a number of turns of the ensemble in the machine
`SCAT`	Produce a set of scatterplots of the beam
`STAT`	Tabulate statistics of the run
`SAVE`	Save the state of the run
`STOP`	Stop the program and return to the operating system

In an interactive run, the input stream is read from the terminal: commands are typed in response to the `ACCSIM >` command prompt. Results of the commands are displayed on the terminal screen and also sent to an auxiliary output file on Unit 2.

In a batch run, the input stream is read from the batch command file, in which the ACCSIM commands are placed in sequence. Results of the commands are stored in the log file for the run.

6.5 Plotting parameters file

This file is used to control the appearance of scatterplots and other plots produced by ACCSIM. It contains parameters used by the GPLOT[33] graph-plotting package, for determining plot scales, axis numbering, etc. The file is not required if no plots are made in a given run. See Section 29 for details of the file format.

6.6 Save file input

The `SAVE` command produces an output file on Unit 11 which describes the complete state of the ACCSIM run at the time the command is given. In a later run, this file can be attached to Unit 12 and read in by giving the `RESTORE` command. This restores the previous run and allows it to be continued from where it left off.

7 Output Files

The standard output files produced by an ACCSIM run are:

Unit 2: Echo output An output file for interactive runs, where the main output goes to the screen.

Unit 6: Main output Displays run parameters and progress.

Unit 7: Plot file Receives scatterplot images for printing.

Unit 9: Injection coordinates Receives the coordinates of each injected macroparticle.

Unit 10: Tabular output Receives a copy of the tabular output produced by the **`TURN`** command.

Unit 11: Save file output Used by the **`SAVE`** command to save the state of a run.

Other units: Auxiliary Many additional files for plotting or post-analysis. These files are summarized in Sections 30 and 31.

7.1 Echo output

This file is normally used only in an interactive run. It receives a copy of all information sent to the terminal screen (Unit 6) and provides a permanent record of the run. In a batch run this record is provided by the log file which is connected to Unit 6. To disable Unit 2, place the following in the batch command file:

```
$ASSIGN/USER NL: FOR002
```

7.2 Main output

This stream provides a complete description of a run and a record of its progress, including the tables and statistics produced when various commands are issued. In interactive mode, this output goes to the terminal screen. In batch mode, it goes to the log file for the batch run. In a batch run, this stream will also receive an echo of all input and calculated parameters for the run. In an interactive run, this display is omitted but can be done at any time by giving the **`PAR`** command.

7.3 Plot file

This stream receives the graphical output for the scatterplots and other plots produced during a run. The data is produced for the specific hardcopy device selected for the run. In an interactive run, a new file is produced for each plot which can be printed immediately by giving the **`HARD`** command (not for GKS metafiles).

In a batch run, all plots are accumulated in one file. Individual plots are separated by "page feed" commands so that the entire file can be printed as a single print job.

7.4 Injection coordinates file

As each macroparticle is injected into the machine, its coordinates at injection are recorded in this file, for reading by other programs. These initial coordinates are not permanently stored internally by the program. See Section 31 for the record structure of this file.

7.5 Tabular record file

The **TURN** command produces a tabular output of particle statistics, painted parameter values and form factors as the turns are executed. This output goes to the main output stream, but an additional copy is sent without headings to this file for reading by other programs. See Section 31 for the record structure of this file.

7.6 Save file

The **SAVE** command produces an output file on Unit 11 which describes the complete state of the ACCSIM run at the time the command is given. In a later run, this file can be attached to Unit 12 and read in by giving the **RESTORE** command. This restores the previous run and allows it to be continued from where it left off.

7.7 Auxiliary Output Files

In addition to the above principal output files, a number of additional files can be produced during a run by giving commands or setting option switches. These files include tracking data for test particles, ensemble coordinate dumps, output of binning arrays, records of beam evolution, and various diagnostic outputs.

The relevant commands, option switches and file formats are documented in Part II of this guide.

8 Preparing a Run

The basic steps in preparing any ACCSIM run are:

1. Decide on location of nodes: rf cavities, collimators, etc.
2. Prepare DIMAD input file for the machine lattice
3. Run DIMAD to get lattice file
4. Prepare main input file
5. Prepare plotting parameters file
6. Prepare command file to assign the I/O streams and run ACCSIM, adding the desired ACCSIM commands if it is a batch run.

7. Submit the command file to the batch queue or execute it at the terminal.

This sounds like a lot of work, but in practice once the files are set up for a given run they can easily be used as "templates" for further runs by editing in the required changes. Typically, the lattice file and the plotting parameters file can be used for a whole sequence of runs.

The following is a representative command file for a **batch** run. See the Appendix for a completely documented example run.

```
$SET DEFAULT disk:[dir]           ! go to working directory
$ASSIGN/USER A121.DAT FOR001      ! main input file
$ASSIGN/USER NL: FOR002           ! disable echo output
$ASSIGN/USER A3.DMO FOR003        ! lattice file
$ASSIGN/USER A121.PLT FOR007      ! plot file
$ASSIGN/USER A.GPL FOR008         ! plotting parameters file
$ASSIGN/USER A121.INJ FOR009      ! injection coordinates file
$ASSIGN/USER A121.REC FOR010      ! tabular record file
$ASSIGN/USER A121.SAV FOR011      ! save file
$ASSIGN/USER A121.TRK FOR021      ! tracking data file
$ASSIGN/USER A121.DMP FOR022      ! dump file
$RUN ACCSIM
TRACK                             ! enable test particle track o/p
TURN 4000 100                     ! execute 4000 turns
SAVE                              ! save state of machine
DUMP                              ! dump ensemble coordinates
TURN 12000 100                    ! execute 12000 turns
SAVE                              ! save state of machine
DUMP                              ! dump ensemble coordinates
EMIT                              ! calculate beam emittances
STAT                              ! print out run statistics
STOP                              ! exit program
```

Note that Unit 6, the main output stream, does not have to be assigned since it goes by default into the log file for the batch run. Once the run is complete, the log file and the plot file can be printed on the appropriate devices.

For an **interactive** run, the following command file could be used to start the program, after which commands would be typed at the terminal.

```
$SET DEFAULT disk:[dir]              ! go to working directory
$ASSIGN/USER A121.DAT FOR001         ! main input file
$ASSIGN/USER A121.OUT FOR002         ! echo output to file
$ASSIGN/USER A3.DMO FOR003           ! lattice file
$ASSIGN/USER A121.PLT FOR007         ! plot file
$ASSIGN/USER A.GPL FOR008            ! plotting parameters file
$ASSIGN/USER A121.INJ FOR009         ! injection coordinates file
$ASSIGN/USER A121.REC FOR010         ! tabular record file
$ASSIGN/USER TT: SYS$INPUT           ! connect terminal to input stream
$RUN ACCSIM
```

In this example, the main output will go to the terminal screen and also to the echo output file **A121.DAT**.

Note that any output streams that are not needed for the run can be discarded by assigning them to the null device. For example,

```
$ASSIGN/USER NL: FOR009
```

will discard the injection coordinates output.

9 Injected Beam

In the following sections the methods used to simulate injected beam distributions and painting schemes are described.

9.1 Distributions

Distributions are generated separately in horizontal, vertical and longitudinal coordinate systems without explicit correlations between planes. Note, however, that the horizontal distribution is specified in betatron phase space where the effect of dispersion is not included. The effect of dispersion in the injection line, and of course in the machine, is calculated by ACCSIM and can result in some horizontal–longitudinal correlation of the beam.

Two methods are used to generate distributions: a simple "masked rectangle" method to generate uniform distributions and a general method to generate a range of distribution profiles. Both these methods utilize the standard uniform $[0, 1]$ random number generator.

The first method generates distributions with *constant density in phase space.* Assume that ΔX and $\Delta X'$ are the horizontal beam half-widths. Generate random numbers R_1 and R_2 in $[0, 1]$ and calculate the deviations

$$\Delta x = (2R_1 - 1)\Delta X,$$

$$\Delta x' = (2R_1 - 1)\Delta X'.$$

This yields a uniform distribution in a rectangle. To get a uniform distribution in an *upright* ellipse with the given half-widths, we calculate the test quantity

$$t = \frac{\Delta x^2}{\Delta X^2} + \frac{\Delta x'^2}{\Delta X'^2}.$$

If $t \leq 1$, the particle lies in the ellipse and we accept it. If $t > 1$, the particle is rejected and a new particle is generated. To generate a *tilted* ellipse with Twiss parameters α, β, γ and ε, we use the bounding rectangle given by

$$\Delta X = \sqrt{\varepsilon\beta}$$

$$\Delta X' = \sqrt{\varepsilon\gamma}$$

and generate random deviations Δx, $\Delta x'$ as above. The test quantity for containment in the ellipse is

$$t = \gamma\Delta x^2 + 2\alpha\Delta x\Delta x' + \beta\Delta x'^2.$$

The particle is accepted if $t \leq \varepsilon$ and rejected if $t > \varepsilon$. These "masking" methods work with reasonable efficiency provided the ellipse area is not too small compared to the rectangular area.

A more general algorithm for generating distributions uses the binomial distribution method described by W. Joho[6]. The shape of the binomial distribution is governed by a single parameter m. Varying m will yield various common distribution types, approaching a Gaussian as $m \to \infty$. For finite m the distribution has a finite range and is therefore preferable to a Gaussian for simulation purposes.

As above, the beam ellipse is specified by the Twiss parameters α, β, γ and ε. If ε specifies the *limiting ellipse* that contains 100% of the beam, then the bounding rectangle is given by

$$\Delta X = \sqrt{\varepsilon\beta}$$

$$\Delta X' = \sqrt{\varepsilon\gamma}$$

as above. Alternatively, we can specify ε for the 2σ *ellipse* that contains 2 standard deviations (in each dimension) of the beam. In this case, the bounding rectangle is given by

$$\Delta X = \sqrt{\frac{m+1}{2}}\sqrt{\varepsilon\beta}$$

$$\Delta X' = \sqrt{\frac{m+1}{2}}\sqrt{\varepsilon\gamma}$$

where m is the binomial distribution parameter. Given random numbers R_1 and R_2 in $[0, 1]$ we generate a point in the ellipse as follows:

$$\begin{aligned} a &= \sqrt{1 - R_1^{1/m}} \\ b &= 2\pi R_2 \end{aligned}$$

$$\begin{aligned} u &= a\cos b \\ v &= a\sin b \\ \Delta x &= u\Delta X \\ \Delta x' &= (u\sin\chi + v\cos\chi)\Delta X' \end{aligned}$$

where χ is the ellipse tilt parameter given by

$$\begin{aligned} \cos\chi &= \sqrt{\frac{1}{1+\alpha^2}} \\ \sin\chi &= -\alpha\cos\chi . \end{aligned}$$

The characteristics of this distribution are tabulated in detail in Reference [6]. The following summary shows, for some common values of m, the distribution shape, the profile shape (1-d projected distribution), and the amount of beam that lies outside the 2σ ellipse. *Note:* the distribution types refer to a single plane with no correlation assumed between vertical and horizontal planes. The physical cross-section of the injected beam is thus rectangular.

m	Distribution	Density $\rho(u,v)$ $(a^2 \equiv u^2+v^2)$	Profile	% of beam outside 2σ ellipse
0.0	Hollow shell	$\frac{1}{\pi}\delta(1-a^2)$	$\frac{1}{\pi}(1-u^2)^{-0.5}$	0
0.5	Flat profile	$\frac{1}{2\pi}(1-a^2)^{-0.5}$	$\frac{1}{2}$	0
1.0	Uniform	$\frac{1}{\pi}$	$\frac{2}{\pi}(1-u^2)^{0.5}$	0
1.5	Elliptical	$\frac{3}{2\pi}(1-a^2)^{0.5}$	$\frac{3}{4}(1-u^2)$	8.9%
2.0	Parabolic	$\frac{2}{\pi}(1-a^2)$	$\frac{8}{3\pi}(1-u^2)^{1.5}$	11.1%
$\to\infty$	Gaussian	$\frac{1}{2\pi\sigma\sigma'}\exp\left(-\frac{x^2}{2\sigma^2}-\frac{x'^2}{2\sigma'^2}\right)$	$\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{x^2}{2\sigma^2}\right)$	12.8%

9.2 Injection coordinates

The methods described in the previous section are used in the same way to generate distributions in the horizontal, vertical, and longitudinal planes, although in the longitudinal plane the beam ellipse is assumed to have no tilt and is always specified by half-widths. The procedures generate for each particle a set of displacements $(\Delta x, \Delta x', \Delta y, \Delta y', \Delta\phi, \Delta(\Delta E))$ from the current location of the injected beam center $(x_0, x_0', y_0, y_0', \phi_0, \Delta E_0)$. These displacements are added to the center coordinates to obtain the actual injection coordinates $(x, x', y, y', \phi, \Delta E)$ for the particle.

> Note: the program does not account for any energy drop due to stripping of H^- or H^0 at injection. Both the injection-line central energy and particle injection energy are assumed to refer to protons.

9.3 Dispersion

The effect of horizontal dispersion in the injection line will also be included. Suppose η_{x0} and η_{x0}' are the injection-line dispersions at the point of injection. If a particle is injected with energy coordinate ΔE then by definition it has total energy $E = E_s + \Delta E$, where E_s is the synchronous or nominal energy of the ring. Thus it has a deviation of $E - E_0$ from the central energy E_0 of the injected beam, which is assumed to be the nominal energy of the injection line (note that, due to energy painting, the injection line may not have the same nominal energy as the ring!). This deviation corresponds to a momentum deviation

$$\frac{p - p_0}{p_0} = \frac{1}{\beta_0^2} \frac{E - E_0}{E_0}$$

where p_0 and β_0 are also nominal values for the injection line. Therefore, the horizontal injection coordinates for the particle are modified by adding the dispersion component:

$$\begin{aligned} x &\longrightarrow x + \eta_{x0} \frac{p - p_0}{p_0} \\ x' &\longrightarrow x' + \eta'_{x0} \frac{p - p_0}{p_0}. \end{aligned}$$

The injection-line dispersions can be specified in the main input file. Alternatively, a flag DMATCH can be set to match the injection-line dispersion to the ring dispersion.

9.4 Painting programs

Injection painting can be achieved in two ways:

1. By altering the closed orbit (x_{co}, y_{co}) at the injection point (e.g. by using bump magnets). This is treated by the program as an instantaneous translation of coordinates before and after the injection point.

2. By altering central values $(x_0, y_0, \phi_0, \Delta E_0)$ of the injected beam (e.g. by using steering or rf elements).

In both cases, the program will alter the relevant parameter value as a function of time. This is controlled by a *painting program* that is specified as a piecewise linear function of time in milliseconds. Each function is determined by a set of time values and corresponding parameter values, read from the main input file. For example, the following set of values

```
TIME=  0     1.3   1.6   4.52  4.82  9.3   9.6  16.0
VALUE= 7.89  7.89  5.26  5.26  2.63  2.63  0.0   0.0
```

will yield the painting function shown in Figure 3.

Up to 100 points can be specified for each painting function. The time values should begin at 0 ms and end at the largest time contemplated for the run. If the actual elapsed simulation time exceeds the last time value, the painting function remains flat at its last value in the list.

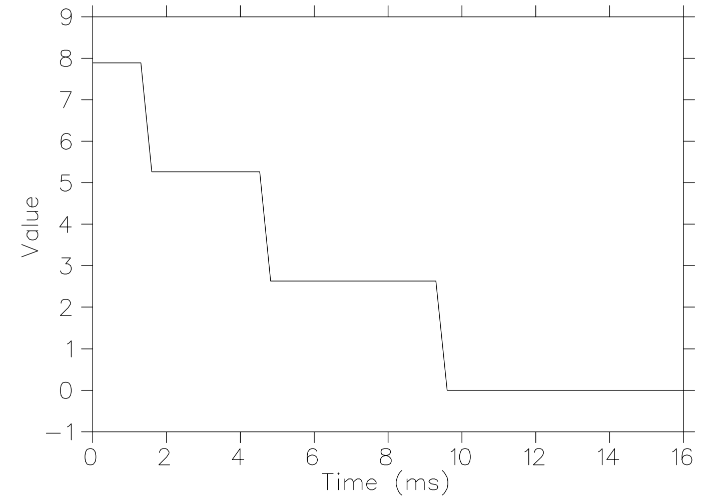

Figure 3: Example painting function

10 Macroparticle Injection

A complete "turn" in ACCSIM consists in following the ensemble of macroparticles from the injection point around the ring and back to the injection point again. Multiturn injection is simulated by adding new macroparticles to the circulating ensemble as turns are executed. These particles are randomly distributed according to the injected beam distributions and are offset according to injected beam center, which may change from turn to turn due to painting.

One exception to this is the *reference particle*, which is injected at the (initial) center coordinates $(x_0, x'_0, y_0, y'_0, \phi_0, \Delta E_0)$ of the injected beam. This particle is designated as Particle 1 and is automatically injected at the beginning of the run. Subsequent randomly-generated particles are numbered in order of their injection into the machine. If desired, the coordinates of the reference particle or any other particle can be changed at any time after its injection by using the **SET** command. The reference particle is the default particle for tracking data file output enabled by the **TRACK** command.

The rate of macroparticle injection is controlled by parameters in the main input file (see Section 26.4). A fixed number of particles can be added on each turn, or, if the number of turns is larger than the total number of particles wanted, a single particle can be added at intervals during the run. In the latter case, a problem can arise. Depending on the betatron tunes, injecting particles at fixed turn intervals may result in a "strobing" effect where particles accumulate in discrete clumps rather than being distributed continuously throughout the phase space. For this reason, particles are injected on randomly- selected turns. Suppose n_p particles are to be injected over n_t turns, and the injection rate n_p/n_t is less than 1. On each turn, a random number $R \in [0, 1]$ is generated and a particle is injected if $R < n_p/n_t$. This results in a certain amount of statistical fluctuation in the injection rate but eliminates any strobing effects.

For the purposes of space charge and other intensity-dependent effects, the actual number of real protons equivalent to the macroparticle population must be known. This is determined by the program using the input parameters NMACRO and PREAL specified in the main input file.

11 Injection Foil

In H^- injection the effect of multiple traversals of the stripping foil by the circulating protons is of great importance for the injection system design. In ACCSIM, the effects of Coulomb scattering and of energy loss in the foil are modelled by Monte Carlo processes.

Nuclear scattering events are relatively rare and are not simulated but the program calculates the expected number of such events during the run. Assuming that each such event leads to particle loss by wide-angle scattering, the resultant fractional beam loss can be estimated.

11.1 Coulomb scattering

In order to choose the appropriate scattering model we first determine the average number m of Coulomb interactions during the passage of a proton through the foil. Following Jackson[7] and others, we use the Rutherford scattering law modified to account for small-angle scattering. To do this, a cutoff angle $\theta_{\rm min}$ is introduced, yielding a differential cross section

$$\frac{d\sigma}{d\Omega} = \left(\frac{2zZe^2}{pv}\right)^2 \frac{1}{(\theta^2+\theta_{\rm min}^2)^2} \tag{1}$$

where z and Z are the atomic numbers of the incident particle and the scattering medium, e is the elementary charge, and p and v are the incident momentum and velocity. The cutoff angle is based on a Fermi-Thomas model for which the atomic radius is

$$a = \mu a_0 Z^{-1/3},$$

where $a_0 = 0.52917706 \times 10^{-8}$cm is the Bohr radius. Here we have introduced a parameter μ to indicate a constant variously given as 1.4 (Jackson) or 0.885 (others), but which we allow to be set by the user. The resulting cutoff angle is given by

$$\theta_{\rm min} = \frac{\hbar}{pa}.$$

Setting $z = 1$ for protons and integrating (1) yields the total cross section

$$\sigma_T = 4\pi\left(\frac{Ze^2}{pv}\right)^2 \frac{1}{\theta_{\rm min}^2}. \tag{2}$$

Knowing the total cross section we can derive the probability density function

$$f(\theta) = \frac{1}{\sigma_T}\frac{d\sigma}{d\theta} = \frac{2\theta_{\rm min}^2\theta}{(\theta_{\rm min}^2+\theta^2)^2} \tag{3}$$

which describes the distribution of angles θ for a single scatter.

If A is the atomic weight and ρ is the density of the foil material then the number of target atoms per unit volume is $N = N_A\rho/A$ where N_A is Avogadro's number. It follows that for a given foil thickness t the average number m of single scattering events for a proton passing through the foil is given by

$$m = Nt\sigma_T.$$

In ACCSIM we wish to generate the distribution of the net angular displacement acquired by a macroparticle undergoing a series of scatters in the foil. Kaminsky *et al.*[8] have noted that for $m \geq 20$ this distribution can be described by Molière's formula for multiple scattering, whereas for $m < 20$ (true for typical injection foils) we are in the "plural" scattering regime where distributions tabulated by Keil *et al.*[9] can be used. As an alternative to the Keil approach, a method based on repeated single scatters can be used. These program options are described in the following sections.

11.2 Multiple scattering

If $m \geq 20$, a standard library routine MLRL is used to generate scattering angles distributed according to the Molière formula. This is a fast routine adapted by C. Kost from the CERN Library routine MLR[10]. An initial call to MLRL is made to initialize its internal tables, and then entry MLRLN is called for each traversal to return a scattering angle in radians. This angle is then randomly projected into angular kicks $\delta x'$ and $\delta y'$ applied to the particle.

11.3 Plural scattering

Keil, Zeitler and Zinn[9] have derived an approximate analytical expression for the distribution of net scattering angles after a foil traversal. This involves a "folding" together of the distribution of single-scattering angles, given by Equation (3) above, with the distribution of the number of single-scattering events per foil traversal. The latter is described by a Poisson distribution with mean m:

$$P(n;m) = \frac{m^n e^{-m}}{m!}. \tag{4}$$

The results of this analysis have been calculated and tabulated in Reference [9] for various values of m. In ACCSIM the tables of the cumulative distribution function $\hat{G}^*(m,\vartheta)$ have been incorporated. ϑ is a normalized angle defined by $\vartheta = \Theta/\theta_{\min}$, where Θ is the net scattering angle and $\theta_{\min}$ is the cut-off angle described above. These tables are given for integral values of m between 1 and 20, and for normalized angles between 0 and 20.

ACCSIM calculates m for the foil to be simulated and does a linear interpolation between the tabulated values to obtain a working distribution table of $\hat{G}$ for the given m. For each foil traversal, a random number is generated and linear interpolation in the table is used to find a corresponding angle value. This generates appropriately-distributed angles.

Since the tabulated distributions extend only up to angles $20 \times \theta_{\min}$, they under-estimate large angle scattering. To remedy this, an auxiliary program KEIL has been written which calculates and tabulates the distributions to arbitrarily large angles. If desired, this program can be used to refine ACCSIM's data base of distribution values.

11.4 Single scattering

As an alternative to the above analytical method for plural scattering, ACCSIM also provides a method which simulates repeated single-scattering events as each particle traverses the foil. This method is adapted from work of A. Thiessen[12].

The number of single scatters per foil traversal has a Poisson distribution with mean m. It follows[13] that the distance travelled between scatters has an exponential distribution. This can be seen by considering that $m = N\sigma_T t$ is proportional to the distance t travelled through the foil material. The constant $t_s = t/m$ is the "mean free path" or average distance between scatters. Considering t as a free variable, Equation (4) gives the probability of n scatters in distance t:

$$p(n; t/t_s) = \frac{(t/t_s)^n e^{-t/t_s}}{n!}.$$

The probability that no scattering events occur in distance t is

$$p(0; t/t_s) = e^{-t/t_s},$$

so the probability that the first scatter occurs in a distance $< t$ is

$$H(t) = 1 - e^{-t/t_s}. \tag{5}$$

This is the cumulative distribution function for the distance between scatters, and the corresponding density function is the exponential distribution

$$h(t) = \frac{1}{t_s} e^{-t/t_s}$$

which has mean t_s as expected.

Using Equation (5), the desired distribution can be generated from uniformly-distributed random numbers $R \in [0, 1]$ by calculating distances

$$d = t_s(-\ln R).$$

For a given particle hitting the foil, we calculate the distance d_1 to the first scatter. We then apply the single-scattering distribution to generate a scattering angle. Then the distance d_2 to the next scatter is calculated, a new scattering angle generated, and so on, until the accumulated distance exceeds the foil thickness, i.e. the particle has exited the foil. For some particles the first distance d_1 will exceed the foil thickness, meaning that the particle did not scatter at all.

As the foil is traversed the scattering angles are randomly projected into the horizontal and vertical planes and a vector sum is accumulated, yielding the net angular displacements $\delta x'$ and $\delta y'$ to be applied to the particle.

To generate the single-scattering angles we first obtain the cumulative distribution function

$$F(\theta) = \int_0^\theta f(x)dx = \frac{\theta^2}{\theta_{\min}^2 + \theta^2}.$$

This function is invertible so that for random numbers $R \in [0, 1]$ we can generate scattering angles θ by

$$\theta = \theta_{\min}\sqrt{\frac{R}{1-R}}.$$

It is seen that this single-scatter model has the advantage of being analytically tractable and does not require any look-up tables or interpolation schemes. It has been found to agree closely with Keil's tabulated results, as one would expect since it is based on the same fundamental assumptions. Moreover, it can be considered more accurate at large angles since no explicit upper limit on the single-scatter angle or total angle has to be imposed.

11.5 Scattering cross sections

The approximate treatment of Coulomb scattering used above is compared by Butler *et al.*[14] to more accurate calculations of differential cross sections using the Hartree-Fock method. It was found that adjusting the effective atomic radius parameter μ would give varying goodness of approximation over the small, intermediate and large angle ranges. This is shown in Figure 4 reproduced from Reference [14].

The value $\mu = 1.15$ gives the best overall fit, whereas $\mu = 1.35$ provides the best approximation at small and large angles. This can be kept in mind when setting the parameter for ACCSIM runs. For reference the curve for $\mu = 0.885$ is also shown since this value is often used in elementary treatments.

The total cross section σ_T and the resulting mean number of scatters m are also only approximately predicted by Equation (2); indeed they vary with parameter μ. Therefore, a more accurate total cross section, such as those tabulated in References [14] and [15], can optionally be input to the program. This cross section will be used only to calculate m, and hence the distribution of the number of scattering events per traversal, whereas the the single-scattering angle distribution will still be determined by parameter μ.

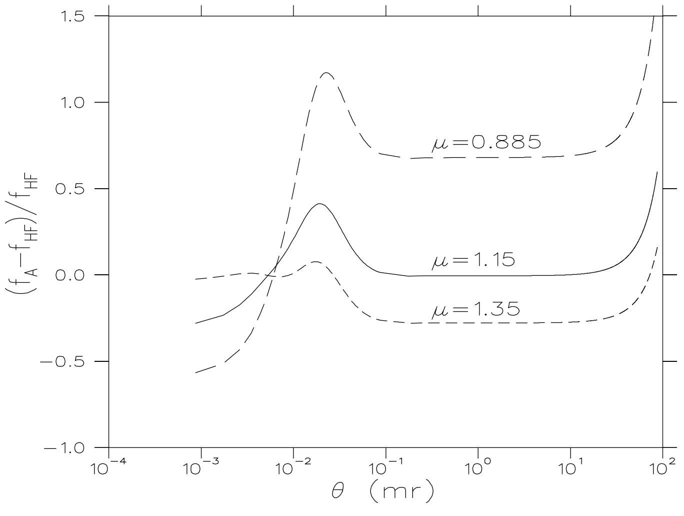

Figure 4: Difference between distributions for cut-off Rutherford model (f_A) and Hartree-Fock (f_{TF}) model for 250μg/cm^2 carbon[14].

11.6 Energy loss

This part of the code was originally developed by D. Raparia[16]. For our scattering system, we compute an energy loss parameter κ:

$$\kappa = \xi / E_{\max}$$

where

$$\xi = \frac{0.30058 \cdot Z m_e c^2 t}{\beta^2 A} \text{ MeV.}$$

In the above, $m_e c^2$ is the electron mass expressed in MeV, t is the foil thickness in g/cm^2, $\beta = v/c$ is the incident particle velocity, and Z and A are the atomic number and atomic weight of the scattering medium. $E_{\max}$ is the maximum energy transferable to an atomic electron, given by:

$$E_{\max} = 2 m_e c^2 \beta^2 \gamma^2 \left[1 + 2\gamma \frac{m_e}{m_p} + \left(\frac{m_e}{m_p} \right)^2 \right]^{-1}$$

where m_p is the proton rest mass.

For $\kappa \le 0.01$ the energy loss can be described by the Landau distribution, whereas for $\kappa > 0.01$ the Vavilov distribution must be used. Typical injection foils are well within the Landau range. For example, with an incident energy of 450 MeV and a foil thickness of 250 μg/cm^2 we have $\kappa \simeq 3 \times 10^{-5}$.

We utilize the CERN library routine DINLAN[11] which computes the inverse of the Landau cumulative distribution function. Given a random number $R \in [0, 1]$, an energy loss δE is generated by the formula

$$\delta E = \langle E \rangle + \xi \left[\text{DINLAN}(R) + 1 + \beta^2 - 0.577216 + \ln \kappa \right].$$

The mean energy loss $\langle E \rangle$ is computed by the Bethe-Bloch formula:

$$\langle E \rangle = \xi \left[\ln \left(\frac{2m_e c^2 \beta^2 \gamma^2 E_{\max}}{I^2} \right) - 2\beta^2 \right],$$

where I is the mean excitation potential of the absorber atom.

The above method also applies to the treatment of energy loss in collimation slits (see Section 14.4) except that, for the thicker material, the Landau distribution is replaced by the Vavilov distribution.

11.7 Nuclear scattering

Nuclear scattering is not directly simulated by ACCSIM since for injection foils it is a relatively rare event and a Monte Carlo treatment gives poor statistics. However, an attempt is made to estimate the losses due to this process.

The total cross section for nuclear scattering during a pass through the foil is estimated from the proton-proton total cross section as

$$\sigma_{\mathrm{n}} \approx \sigma_{\mathrm{pp}} A^{\frac{2}{3}} \quad \text{(barn)}.$$

If N is the number of target atoms per cm^2 then the probability of a nuclear scattering event in a given traversal is

$$P_n = N\sigma_{\mathrm{n}} \cdot 10^{-24}.$$

If H is the total number of foil traversals for all particles during a run, then the number of particles L lost due to nuclear scattering can be estimated by

$$L \cong P_n H.$$

This is only a good approximation if the number of traversals per particle is small compared to the total number of traversals, i.e. we have a sufficiently large population of particles and the particles do not hit the foil too often.

11.8 Foil histograms

Traversals of the foil by macroparticles are recorded in two histograms. These are 2-dimensional histograms over the foil area and have a user-specified bin size. One array accumulates the number of traversals as a function of position on the foil. The other accumulates the energy deposited as a function of position on the foil. The latter represents a pessimistic estimate since it assumes that all energy lost by a particle during foil traversal is deposited in the foil and none is carried out by secondary particles.

For further analysis these arrays can be written to a file using the **FHIST** command, and the **FOIL** command provides estimates of projected heat distributions and average heating power (see Section 28).

12 Internal Target

In lieu of an injection foil, an internal target can be simulated using essentially the same methods. For thicker materials with $m > 20$, the program will automatically use the multiple scattering routine instead of plural scattering methods. The dimensions, thickness and material properties of the target are specified in the same way as for a foil.

Due to program logic, the internal target cannot be arbitrarily placed but must reside at the same location as a foil: at the start of the lattice. If necessary this can be accomplished by a re-arrangement of the beam-line definitions for the DIMAD run.

13 Inter-node Transport

The transport of particles between nodes is accomplished using the first-order transfer matrices generated by DIMAD. In order to correctly describe the phase of particles with respect to the rf system, an extension to the conventional TRANSPORT formalism is required. This is described in detail in Reference [3], of which we summarize the results in this section.

The longitudinal positional coordinate ℓ used in the TRANSPORT matrix refers to the path length difference between an arbitrary particle's orbit and the reference orbit followed by the synchronous particle. On the other hand, the rf phase coordinate ϕ is determined by the arrival time of particles at the accelerating gaps, which is a function of particle velocity as well as path length.

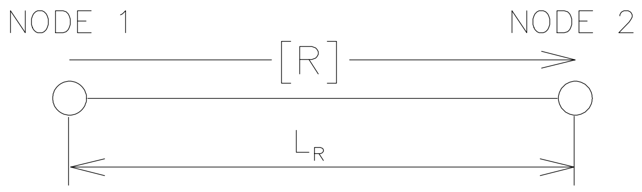

Figure 5: Transport between nodes.

Suppose that a particle has arrived at Node 1. Before proceeding to Node 2 it may receive kicks simulating an rf cavity, thin lens, or space-charge update at Node 1. Assume that, after any such kicks are applied, its coordinates are $x_1, x_1', y_1, y_1', \ell_1, \delta_1$. If R is the transfer matrix from Node 1 to Node 2, then the particle's path length difference at node 2 will be

$$\ell_2 = \ell_1 + R_{51}x_1 + R_{52}x_1' + R_{56}\delta_1.$$

If the particle's rf phase at Node 1 is ϕ_1 then, to first order, its phase at Node 2 is given by

$$\phi_2 = \phi_1 + \frac{2\pi h}{L_T}\left[(\ell_2 - \ell_1) - \frac{L_R}{\gamma_s^2}\delta_1\right]$$

where h is the harmonic number, L_T is the total length of the machine, L_R is the length of the reference orbit between the nodes, and γ_s is the synchronous energy.

Incorporating the above with the transformations for the other coordinates we obtain the following set of transport equations for the coordinates at Node 2:

$$\begin{aligned}
x_2 &= R_{11}x_1 + R_{12}x_1' + R_{16}\delta_1 \\
x_2' &= R_{21}x_1 + R_{22}x_1' + R_{26}\delta_1 \\
y_2 &= R_{33}y_1 + R_{34}y_1' \\
y_2' &= R_{43}y_1 + R_{44}y_1' \\
\phi_2 &= \phi_1 + \frac{2\pi h}{L_T}\left[R_{51}x_1 + R_{52}x_1' + \left(R_{56} - \frac{L_R}{\gamma_s^2}\right)\delta_1\right] \\
\delta_2 &= \delta_1.
\end{aligned}$$

Note that for the purposes of ACCSIM we have eliminated matrix terms that are assumed to be zero, including the x–y coupling terms (transverse coupling can, however, be introduced using thin skew quadrupoles).

The coordinate $\delta = \Delta p/p_s$ is not actually stored by the program but is converted from the stored coordinate $\Delta E/E_s$ using the linear relation

$$\frac{\Delta p}{p_s} = \frac{1}{\beta_s^2}\frac{\Delta E}{E_s}.$$

This is based on expansion to first order about the synchronous coordinates and is therefore a valid approximation in the context of first-order transport. The use of ΔE as a working coordinate makes it straightforward to integrate into the tracking process the effects of rf cavities, longitudinal space charge and foil energy loss.

14 Node Properties

The effects of rf cavities, thin lenses, etc., that are found at the various node locations, are treated as impulsive "kicks" applied to the particle coordinates. Nodes can also be used as aperture points or monitor points where particle coordinates can be checked for loss or output as files or scatterplots. Each node can have any of the following properties:

- RF cavity
- Longitudinal space-charge update
- Thin lens
- Aperture or collimation slit
- Plotting or tracing output

These node properties are discussed in detail in the following sections.

14.1 RF cavity

As noted by Hereward[17], the longitudinal motion due to rf cavities can be calculated using a standard set of difference equations. If there are N_c equally-spaced cavities with total rf voltage V then the transformation of coordinates from cavity n to cavity $n+1$ is

$$\Delta E_{n+1} = \Delta E_n + \frac{eV}{N_c}\sin\phi_n \tag{6}$$

$$\phi_{n+1} = \phi_n + \frac{2\pi\eta_0 h}{N_c\beta_s^2 E_s}\Delta E_{n+1} \tag{7}$$

where e is the elementary charge, h is the harmonic number, and E_s is the total synchronous energy. Note that we assume E_s = constant and $\phi_s = 0$ since ACCSIM does not at present simulate beam acceleration. In general, a refinement of Hereward's equations is required[18,19] in order to correctly treat acceleration.

The parameter η_0 is the frequency slip factor given by

$$\eta_0 = \alpha_p - \frac{1}{\gamma_s^2} = \frac{1}{\gamma_t^2} - \frac{1}{\gamma_s^2}.$$

This quantity relates a particle's revolution period to its momentum deviation and consists of the "distance" term α_p and the "velocity" term γ_s^{-2}, the two terms becoming equal at transition.

Equation (6) describes the energy kick due to passage through the cavity. In ACCSIM this equation is applied at each node where a cavity resides. Equation (7), however, is not applied since we already have a transport equation for the change in ϕ between nodes (see Section 13). Comparing this transport equation with Equation (7) will reveal that the transport equation, which refers to arbitrary locations in the lattice, is essentially a more general version of (7). This can be seen by noting that

$$\alpha_p = \frac{R_{51}\eta + R_{52}\eta' + R_{56}}{L_T}$$

$$\delta = \frac{1}{\beta_s^2}\frac{\Delta E}{E_s}$$

where R is the full-turn transfer matrix, so that for a single cavity ($N_c = 1$, $L_R = L_T$) equation (7) takes the form

$$\phi_{n+1} = \phi_n + \frac{2\pi h}{L_T}\left[R_{51}\eta\delta + R_{52}\eta'\delta + \left(R_{56} - \frac{L_T}{\gamma_s^2}\right)\delta\right].$$

The transport equation is actually more accurate, since it takes into account the effect of betatron amplitude on path length,

$$R_{51}(x - \eta\delta) + R_{52}(x' - \eta'\delta),$$

whereas Equation (7) neglects this.

14.1.1 Rf harmonics

Program options are provided for adding one or two additional harmonic components to the rf system. These are specified by voltages V_1 and V_2, harmonic numbers h_1 and h_2, and phases ψ_1 and ψ_2, so that the total rf voltage is

$$V = V_0 \sin\phi + V_1 \sin h_1(\phi + \psi_1) + V_2 \sin h_2(\phi + \psi_2).$$

See Section 26.2 for information on how to set these parameters.

14.2 Longitudinal space charge

The effect of longitudinal space charge is simulated by a series of energy kicks at designated nodes. The usual place to apply such kicks is at the cavities. This constitutes a modification of Equation (6):

$$\Delta E_{n+1} = \Delta E_n + \frac{eV}{N_c}\sin\phi_n + \frac{eV_{\rm sc}}{N_c} \tag{8}$$

where $V_{\rm sc}$ is the total longitudinal space charge potential in volts per turn.

The energy gain due to space charge is applied impulsively whereas in reality the space charge potential acts continuously as the particle moves to the next cavity. Thus the rf phase change between cavities will be somewhat over- or underestimated. This, however, is not a bad state of affairs since it renders the equations stable. As discussed by Koscielniak[18,19], the equation pair (8,7), which he states in a more general version, is a symplectic mapping and constitutes a first-order *canonical integrator*. In his code LONG1D he uses a more accurate second-order integrator where basic integration step is divided into two parts and the space-charge potential is calculated and applied at the mid-point. In ACCSIM, such a second-order scheme can be implemented by placing the space-charge update (kick) points halfway between the cavities instead of at the cavities. This, however, increases the number of nodes and is usually not essential: the differences in phase-space trajectory and oscillation frequency between first- and second-order results are rather small and could be considered insignificant in light of other approximations being made in the program.

As in many other codes, the space charge potential is evaluated by binning the ensemble of particles with respect to their rf phase coordinate ϕ, in order to obtain the *line density* $\lambda(s)$ or number of particles per unit distance along the bunch. Assuming a perfectly conducting smooth wall, the on-axis longitudinal space-charge field[20,21] is related to the line density gradient by:

$$E_{\rm sc} = -\frac{e}{4\pi\varepsilon_0}\frac{g_0}{\gamma_s^2}\frac{d\lambda}{ds}$$

where e is the elementary charge, ε_0 is the permittivity of free space, and g_0 is the "geometric factor" based on the ratio b/a of vacuum chamber radius to beam radius:

$$g_0 = 1 + 2\ln\frac{b}{a}.$$

Off-axis particles experience a slightly smaller longitudinal space-charge field[30]. This correction is covered in Section 15.2.

A particle whose local density gradient is $d\lambda/ds$ will experience a space-charge potential, expressed in volts per turn, of

$$V_{\rm sc} = 2\pi R E_{\rm sc} = -\frac{e}{2\varepsilon_0}\frac{g_0 R}{\gamma_s^2}\frac{d\lambda}{ds}$$

where $R = L_T/2\pi$ is the average machine radius.

Particles are binned and $V_{\rm sc}$ is calculated when the ensemble arrives at each node designated as a longitudinal space-charge update node. Indeed, it is a general rule that $V_{\rm sc}$ must be re-calculated every time the energy kicks are applied, otherwise the integrating equations do not work as advertised and non-physical emittance growth results. Since $V_{\rm sc}$ represents the total voltage per turn, the actual kick applied at a node is calculated as:

$$\Delta E \longrightarrow \Delta E + eV_{\rm sc}\frac{L_{\rm sc}}{L_T}$$

where $L_{\rm sc}$ is the distance to the *next space-charge update node.*

To obtain the line density the particles are binned in their rf phase coordinate, which is related to longitudinal distance by

$$\Delta\phi = -\frac{h}{R}\Delta s.$$

Here, the sign change indicates that particles that are farther ahead in distance will have more negative rf phase. Thus the line density λ is obtained from the phase density, say λ^* by

$$\lambda(s) = \frac{h}{R}\lambda^*(\phi).$$

Viewing ϕ as a function of s we have

$$\frac{d\lambda}{ds} = \frac{h}{R}\frac{d}{ds}\lambda^*(\phi(s)) = \frac{h}{R}\frac{d\lambda^*}{d\phi}\frac{d\phi}{ds} = -\frac{h^2}{R^2}\frac{d\lambda^*}{d\phi}$$

which allows us to find the line density gradient from the phase density gradient.

To accomplish the binning we define a series of N equally spaced node points ϕ_k ranging from $-\pi$ to π, with spacing $\Delta\phi$ between the node points. The kth node point has value $\phi_k = -\pi + (k-1)\Delta\phi$. To get a phase distribution we use a "weighted assignment" method, which gives a smoother distribution than simple binning. With each node point is associated a count f_k. For a particle of phase ϕ we find the closest node point $\phi_k \le \phi$ and compute a weight factor

$$w = 1 - \frac{\phi - \phi_k}{\Delta\phi}.$$

The weight w is added to the count f_k while $1 - w$ is added to the count f_{k+1}, so that the total count for the distribution is equal to the number of macroparticles.

From the frequency function $f(k) = f_k$, defined at each node point, we need to estimate a derivative df/dk using finite-difference formulas. This poses a problem since statistical fluctuations in the phase distribution can result in wild variations in the derivative. Wei *et al.*[22] have derived the criterion that, as the number of bins is increased in order to provide an adequate description of the bunch structure, the number of particles must increase as the cube of the number of bins in order to keep the same level of statistical noise. Obviously this cannot be satisfied by ACCSIM where the number of particles must be increased from zero to the maximum over the course of a run. The consequence of this is that individual particle motion in small ensembles will not be accurate: particles will exhibit longitudinal amplitude variations that are sensitive to the fluctuations in the calculated space charge force. This may influence some statistics such as foil traversals or losses at apertures.

These variations are incoherent and do not appear to have much effect on the net distribution of particles. In tests of longitudinal distributions evolving over many turns, the basic characteristics of the distribution are relatively independent of the amount of statistical noise. Noise that is quite prominent in the space-charge potential curve is revealed only as a blurring of the finer details in the distribution.

The problem of statistical noise can be alleviated to some extent by smoothing. If one has several thousand particles in a fairly well-defined bunch, Fourier methods[19,23] can be used to smooth the distribution and obtain its derivative. These methods have proved less than satisfactory for ACCSIM's small ensembles. As an alternative, two digital filtering techniques have been implemented. The first is a five-point averaging filter[24] applied to the f_k:

$$f_k \longrightarrow \frac{1}{5}(f_{k-2} + f_{k-1} + f_k + f_{k+1} + f_{k+2}).$$

To avoid anomalous "tails" being generated at the edges of the distribution, special formulas are used. If ϕ_I is the first node point with a nonzero count, then

$$f_I \longrightarrow \frac{1}{5}(3f_I + 2f_{I+1} + f_{I+2} - f_{I+4})$$

$$f_{I+1} \longrightarrow \frac{1}{10}(4f_I + 3f_{I+1} + 2f_{I+2} + f_{I+3}),$$

with similar formulas being applied at the opposite edge of the distribution. This procedure can be iterated a number of times, as specified in the main input file, to get the desired amount of smoothing. Two such iterations are usually sufficient.

In computing the derivative, we use a five-point formula that also provides simultaneous smoothing[25]:

$$f'(k) \approx \frac{1}{10}(-2f_{k-2} - f_{k-1} + f_{k+1} + 2f_{k+2}).$$

Again, special treatment is used at the edges, recognizing that at the edge of the bunch where the density decreases to zero, there may be a discontinuity in the gradient, so that we should compute it from the appropriate direction:

$$f'(I) \approx \frac{1}{20}(-21f_I + 13f_{I+1} + 17f_{I+2} - 9f_{I+3})$$

$$f'(I+1) \approx \frac{1}{20}(-11f_I + 3f_{I+1} + 7f_{I+2} + f_{I+3}).$$

The formulas at the other edge have the indices and signs reversed.

From the above procedure we get a reasonably smooth estimate of the derivative df/dk. Considering that the density function λ^* is approximated at the kth node point by $f_k/\Delta\phi$, we have

$$\frac{df}{d\phi} = \frac{df}{dk}\frac{dk}{d\phi} = \frac{1}{\Delta\phi}\frac{df}{dk}$$

$$\frac{d\lambda^*}{d\phi} = \frac{d\lambda^*}{df}\frac{df}{d\phi} = \frac{1}{\Delta\phi}\frac{df}{d\phi}$$

If we have N_m macroparticles simulating N_p real particles (N_p/h per bucket) then the phase density gradient in real particles per radian is given by:

$$\frac{d\lambda^*}{d\phi} = \frac{N_p}{hN_m}\frac{1}{(\Delta\phi)^2}\frac{df}{dk}.$$

The longitudinal space-charge update nodes are specified by the array `ILSC` in the main input file (see Section 25). Normally they are placed at the same nodes as the cavities. For high synchrotron frequencies, a more accurate calculation can be made by placing additional update nodes between the cavities.

14.3 Thin lens

Thin lenses up to octupole order can be placed at any node and are treated as angular kicks. Each thin lens is specified by an order n and by an integrated strength K_nL in units of (m^{-n}). The field expansion coefficient K_n is

$$K_n = \frac{1}{B\rho}\left.\frac{\partial B_y}{\partial x^n}\right|_{x=0}$$

where $B\rho$ is the beam rigidity and B_y is the vertical field component for the magnetic element. The supported element types are:

$n = 0$	Dipole
$n = 1$	Quadrupole
$n = 2$	Sextupole
$n = 3$	Octupole

Each element can be placed in "normal" orientation or "skew" orientation, where it is rotated by $\pi/(2n+2)$.

The kicks $\delta x'$ and $\delta y'$ are obtained from the general formulas for normal elements:

$$\begin{aligned}
\delta x' &= \frac{1}{1+\delta}\frac{K_n L}{n!}\,\mathrm{Re}\left\{(x+iy)^n\right\} \\
\delta y' &= \frac{-1}{1+\delta}\frac{K_n L}{n!}\,\mathrm{Im}\left\{(x+iy)^n\right\}
\end{aligned}$$

or for skew elements:

$$\begin{aligned}
\delta x' &= \frac{-1}{1+\delta}\frac{K_n L}{n!}\,\mathrm{Re}\left\{i(x+iy)^n\right\} \\
\delta y' &= \frac{1}{1+\delta}\frac{K_n L}{n!}\,\mathrm{Im}\left\{i(x+iy)^n\right\}
\end{aligned}$$

where $\delta = \Delta p/p_s$ is the fractional momentum deviation.

For example, in the simplest case of a thin dipole the transformations are:

$$x' \longrightarrow x' + \frac{K_0 L}{1+\delta}$$

whereas for a "skew" dipole we get the vertical bend:

$$y' \longrightarrow y' + \frac{K_0 L}{1+\delta}.$$

Coupling can be simulated using a thin skew quad:

$$x' \to x' + \frac{K_1 L}{1+\delta}\,y$$

$$y' \to y' + \frac{K_1 L}{1+\delta}\,x.$$

Non-linearities can be introduced into the machine using the higher-order thin lenses. This approach has the advantage that the thin-lens transformations are not only efficient to calculate but are symplectic, a necessary condition to avoid non-physical behaviour over large numbers of turns.

14.4 Aperture or collimation slit

These two node properties designate locations in the machine where the (x, y) coordinates of particles are to be checked against user-defined limits. Apertures can be horizontal, vertical, rectangular or elliptical. If a particle exceeds an aperture limit, it is tagged as lost and is removed from the circulating ensemble. The particle's coordinates at the point of loss, however, will remain in storage for later analysis or plotting. The turn number and node number of each loss will also be recorded. The loss data can be summarized at any time during a run using the `STAT` and `LOSS` commands.

A collimation slit is considered as a special type of horizontal or vertical aperture. Particles that exceed the aperture are not tagged as lost, but are considered to traverse the slit material. Multiple scattering and energy loss effects are simulated for each such particle. A collimation slit can be used in conjunction with one or more downstream apertures to simulate a collimation system based on an emittance-defining slit followed by downstream absorbers.

The parameters Z, A, ρ, I for the slit material and its thickness t are read from the main input file. The multiple scattering and energy loss processes are modelled by a set of routines analogous to those for the injection foil or internal target. A multiple scattering routine based on Molière theory, and an energy loss routine based on the Vavilov distribution, are applied for each particle traversing the slit material. Sections 11.2 and 11.6 of this document can be consulted for details of these techniques. In the case of energy loss, the treatment is the same as that described in Section 11.6 except that the routine DINLAN (Landau distribution) is replaced by the routine DINVAV (Vavilov distribution).

The nuclear scattering loss in the slit material will also be estimated from the total number of slit traversals, as for the injection foil. This is described in Section 11.7, and similar remarks on accuracy will apply.

14.5 Plot or track output

At any node a flag can be set to enable the generation of scatterplots of the particle coordinates at that node. Scatterplots are produced only on turns that are multiples of the NPLOT parameter specified in the main input file. This is done automatically during the execution of the designated plotting turn. If no other nodes have their plot flags set, then by default a plot will be made at Node 1 (the injection point). Node 1 is always plotted at the *end* of the plotting turn rather than the beginning. For further information on scatterplots and graphics, see Section 18.

An additional flag is provided for output of test particle coordinates at any node. This output is enabled by the **`TRACK`** command, described in Section 28, and is done for a designated test particle. Output takes place on every turn for every enabled node. As with the plotting flag, output at Node 1, the injection point, is enabled by default.

15 Transverse Space Charge

The DQ package of modules for handling transverse space charge was developed by H. Schönauer and is described in his separate document[2]. To briefly summarize, this method derives an effective potential taking into account the Lissajous motion of particles in x—y space during their orbits. The emittances (Courant-Snyder invariants) ε_x and ε_y are calculated for each particle and particles are binned in the 2-dimensional "amplitude space" ε_x—ε_y. From this distribution a transverse space charge potential function is derived and resulting x and y tune shifts are calculated for each amplitude bin. For each particle these tune shifts are weighted by the local longitudinal charge density (line density) for the particle's position in the bunch. The line density is calculated using the same longitudinal binning process described in Section 14.2.

ACCSIM incorporates this space-charge calculation in both "passive" and "active" forms. In the passive form, the calculation is done in order to print out the central tune shifts and other figures of merit for the ensemble at regular turn intervals. In the active form, the calculation is done on every turn and the tune shifts for each particle are calculated and applied to the particle motion. The intent in the latter case is to model only the effect on the particle tunes and not any direct amplitude effects, although indirect amplitude effects can result such as resonant growth caused by tune shift.

The following sections describe the method used to incorporate the space-charge tune shifts into the particle tracking process.

15.1 Tune-shift rotation

The space-charge tune shifts are simulated by rotating the particles back in betatron phase space on a turn-by-turn basis. Given a tune shift ΔQ we essentially wish to shift the betatron phase of a particle by the appropriate amount in a manner that is consistent with the properties of the motion. This is done by deriving an appropriate transformation matrix, with the necessary condition that the transformation be symplectic.

The required phase shift to be achieved is

$$\Delta\mu = 2\pi\Delta Q.$$

In betatron phase space this can be implemented as a rotation given by the matrix

$$\begin{bmatrix} \cos\Delta\mu + \alpha\sin\Delta\mu & \beta\sin\Delta\mu \\ -\gamma\sin\Delta\mu & \cos\Delta\mu - \alpha\sin\Delta\mu \end{bmatrix} = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}$$

where α, β, γ are the lattice function values at the location where the tune shift is to be applied. This matrix is that of a "periodic section"[26]. It preserves the values of α, β, γ and therefore does not alter the Courant-Snyder ellipse at the given location.

The above rotation is calculated and applied for both the horizontal and vertical planes. However, in the horizontal case, the physical coordinates (x, x') must first be transformed to phase-space coordinates (x_b, x'_b):

$$x_b = x - \eta\delta$$

$$x'_b = x' - \eta'\delta$$

where η, η' are the dispersions at the location where the tune shift is to be applied.

15.2 Tune-shift transformations

Since the tune shifts for a given particle are proportional to the line density λ which is a function of the phase coordinate ϕ, some care must be taken to ensure that the tracking process remains symplectic in the presence of the horizontal tune-shift rotation.

Attempts to implement the tune shift as a single impulsive transformation of coordinates exhibited problems with emittance non-conservation. However, this difficulty was resolved by R. Baartman[29], who investigated the motion under the influence of an rf-phase-modulated tune shift and found that a series of distinct steps are required to correctly transform the coordinates:

1. Transform to dispersion-free coordinates. This is accomplished by applying the matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -\eta \\ 0 & 1 & 0 & 0 & 0 & -\eta' \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ -\eta' & \eta & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

   In addition to x and x' the rf phase ϕ is also transformed when passing to the dispersion-free domain. Indeed, this is required by the symplectic conditions

$$R_{11}R_{26} - R_{16}R_{21} + R_{51} = 0$$

$$R_{12}R_{26} - R_{16}R_{22} + R_{52} = 0.$$

   Note, however, that the signs of R_{51} and R_{52} have been inverted. This is for compatibility with the DIMAD matrices. In DIMAD the convention is that positive ℓ means a particle has a longer path length than the reference particle. This results in the signs of the path length terms $(R_{51}, R_{52}, \ldots)$ being opposite to the "canonical" signs implied by the symplectic conditions above.

2. Calculate $\lambda(\phi)$, ΔQ_x, ΔQ_y, and apply the tune-shift rotations:

$$\begin{bmatrix} D_{11} & D_{12} & 0 & 0 & 0 & 0 \\ D_{21} & D_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & D_{33} & D_{34} & 0 & 0 \\ 0 & 0 & D_{43} & D_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

   In ACCSIM, $\lambda(\phi)$ is available from routine LINEDENS which bins particles in ϕ and smooths the resulting density curve (see Section 14.2 for details).

3. Transform back to physical coordinates:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \eta \\ 0 & 1 & 0 & 0 & 0 & \eta' \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \eta' & -\eta & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Algorithmically, the important feature of the above is that the phase ϕ used in calculating the horizontal tune shift is the phase *at the point* where the rotation is applied. Trying to concatenate the above matrices into a single transformation results in an asymmetry and non-conservation of emittance.

Baartman has also pointed out[30] that since ϕ is a canonical variable with conjugate $W = \Delta E/\omega_{\mathrm{rf}}$, in conjunction with step (2) above the particle's energy coordinate should also be altered according to

$$\Delta E \longrightarrow \Delta E + \pi\omega_{\mathrm{rf}} p_s \left(\varepsilon_x \frac{dQ_x}{d\phi} + \varepsilon_y \frac{dQ_y}{d\phi} \right),$$

where ε_x and ε_y are the particle's Courant-Snyder invariants. Physically, this takes into account the variation of longitudinal space charge force with transverse position relative to the beam axis. This is a relatively small effect but has been shown to be important when the tune shift is large. In ACCSIM the quantities $dQ_x/d\phi$ and $dQ_y/d\phi$ can be obtained from $d\lambda/d\phi$ which is calculated from the smoothed line density by the procedure described in Section 14.2. *Note:* This correction has not yet been successfully implemented in the code but is included here for reference.

16 Chromaticity

A program option is also provided for simulating linear chromatic tune-shifts. This is a second-order effect and therefore is not seen when tracking by first-order matrices. This option can be used in conjunction with transverse space charge to simulate particle motion in "tune space"[27].

The first-order chromaticities C_x, C_y of the lattice are calculated by DIMAD and read by ACCSIM from the lattice file. For a given particle of momentum deviation $\delta = \Delta p/p_s$ the chromatic tune shifts are given by

$$\Delta Q_x = C_x \delta, \quad \Delta Q_y = C_y \delta.$$

Similarly to the treatment of space-charge tune shifts, these tune shifts are simulated by applying a symplectic transformation which rotates the particles in betatron phase space by $2\pi\Delta Q_x$ and $2\pi\Delta Q_y$. This is done once per turn, at the end of the turn.

In machines studied thus far with ACCSIM the linear tune-shift relations above have been a good approximation, but higher order terms could easily be added in the code. These terms could be obtained through a DIMAD analysis.

17 Tabular Output

As the ACCSIM run progresses, some statistics and figures of merit are output at regular turn intervals. This output goes to Unit 6 (the screen in interactive mode and the log file in batch mode) and also to an auxiliary file on Unit 2.

The following example shows the first 500 turns of a multiturn injection run:

```
Turn  Time    Nm  Hits avg Lost  Xco    Yco     Y0   DE0  Tsync      Np        Ex    Ey    Bf   Gf     DQx    DQy
 100 0.097    46   136   3    0  0.00   7.89   10.23 0.70 450.00  0.799E+11  49.3  8.7  0.16 0.99  -0.001 -0.006
 200 0.195    97   500   5    0  0.00   7.89   10.23 0.70 450.00  0.169E+12  46.9  8.7  0.14 0.88  -0.002 -0.012
 300 0.292   134  1081   8    0  0.00   7.89   10.23 0.70 450.00  0.233E+12  49.9  9.0  0.15 0.91  -0.003 -0.015
 400 0.389   196  1956  10    0  0.00   7.89   10.23 0.70 450.00  0.341E+12  49.7  8.9  0.15 0.91  -0.005 -0.023
 500 0.487   244  3150  13    0  0.00   7.89   10.23 0.70 450.00  0.424E+12  50.1  9.4  0.15 0.93  -0.006 -0.028
```

Turn is the elapsed number of turns since the beginning of the run. **Time** is the elapsed simulation time in milliseconds. This is the product of the turn number and the synchronous revolution time.

Nm is the total number of macroparticles that have been injected, including lost particles. In the above example, the rate of injection is less than 1 particle per turn. Some statistical fluctuation is seen owing to the random algorithm used to control particle injection (see Section 10).

Hits is the total number of traversals of the injection foil for all particles, and **avg** is the average traversals per particle. If there are N_m particles accumulated and h_i is the number of times the ith particle has hit the foil, then

$$\texttt{Hits} = \sum_{i=1}^{N_m} h_i,$$

$$\texttt{avg} = \texttt{Hits}/N_m.$$

Lost is the total number of particles lost at apertures.

Xco and **Yco** record the coordinates in millimeters of the closed orbit at the injection point. These values can be swept to simulate bump magnets used in painting. Similarly, the injected beam central values **Y0** and **DE0** can be swept to simulate steering or energy ramping of the injected beam.

Tsync is the synchronous, or reference, energy of the ring in MeV. This is at present a constant but in future will be allowed to be ramped to simulate an acceleration cycle.

The remaining columns are printed only if the transverse space charge calculation is enabled (option DQPAR or DQVAR):

Np is the number of circulating protons simulated by the macroparticle ensemble.

Ex and **Ey** are the emittances of the beam in mm-mr. These are based on a beam ellipse containing 95%–100% of the ensemble, depending on the setting of the beam fraction parameter EMFRAC.

Bf is the longitudinal bunching factor $\overline{\lambda}/\hat{\lambda}$, or ratio of average density to peak density. **Gf** is the form factor G[28] for the transverse 2-dimensional distribution. This is the ratio of peak density to average density ranging from < 1 (hollow) through 1 (uniform) and 2 (parabolic). Larger values indicate that the beam has developed a halo.

DQx and **DQy** are the maximum tune shifts for the beam center, calculated from the ensemble by the DQ package.

18 Graphics

The primary emphasis of the built-in graphics is in efficiently generating the phase-space scatterplots which are usually produced at regular intervals during a run. Some other plots can also be produced, but elaborate data presentations are better done by post-processing (see next Section).

The graphics devices to be used in a run are selected by defining logical names. The logical name **TRIUMF_TERMINAL_TYPE** can be set to the following choices:

VT640	Retro-graphics VT640 or compatible terminal (default)
CIT467	C. Itoh 467 colour graphics terminal
VAXSTATION	VaxStation running UIS

In batch mode, scatterplots are not displayed and so the above assignment is not necessary.

The logical name **TRIUMF_HARDCOPY_TYPE** is used to select the hardcopy output device from the following choices:

HPLASER	HP Laserjet printer (default)
LN03	DEC LN03+ laser printer
GKSMETA	GKS Metafile (where supported)

User requirements to date have determined the selection of devices offered. Additional devices can be added as necessary from the range supported by the TRIUMF Graphics Library[33]. This is easily done by modifying the routine **SETUP_GRAPHICS**. Access to additional display and hardcopy devices can also be achieved through EDGR[34] image files and the GKS Metafile output. The latter output requires that ACCSIM be linked with a local GKS library. In most GKS installations, a "replay" program is available which will display or print metafile images on various devices.

In interactive mode, plots are displayed on the screen and simultaneously written to a plot file (Unit 7) in the appropriate format for the chosen hardcopy device. A hardcopy of the plot being viewed can be made immediately using ACCSIM's **HARD** command which queues the plot file for printing. For each plot produced, a new plot file is opened, so that only the most recent plot will be printed by the **HARD** command. (This does not apply to GKS metafile output which accumulates all images in one file and for which the **HARD** command is disabled.)

In batch mode, the scatterplot display is of course disabled, and all plot images are accumulated in a single file attached to Unit 7. The images are separated by "page feed" commands so that the file can be printed as a single job when the run is complete.

19 Pre- and Post-processing

ACCSIM has been developed in a computing environment where a variety of analysis and graphics facilities are available. Of course, DIMAD is the primary pre-processing tool and can be used to design and tune the lattice used in the ACCSIM simulation. Through the auxiliary distribution input on Unit 4, particle distributions from other programs can be used in a simulation. For analyzing and plotting the results of a simulation, a package of TRIUMF programs has proved very useful. This package comprises three programs:

OPDATA [31] is a data analysis and plotting program offering a wide range of operations such as fitting, FFT, integration and differentiation, etc., in an efficient command-driven environment.

PLOTDATA [32] offers similar capabilities to OPDATA with emphasis on data presentation through a comprehensive set of graphing options.

EDGR [34] is a graphical editor allowing device-independent storage, editing, and creation of images. ACCSIM, OPDATA, and PLOTDATA are capable of producing EDGR image files.

As an example of post-processing, the following procedure will produce a "tune spread" scatterplot showing instantaneous betatron tunes for an accumulated ensemble. The space-charge tune shifts for each particle will be read from a file, whereas the chromatic tune shifts will be calculated directly.

In the ACCSIM run, the file FINAL.DAT is attached to Unit 22. At the end of the run, the internal arrays for the particles can be written to this file using the **DUMP** command (see Section 31 for details of the data structure). For space and efficiency reasons the file is written in binary format. Such a file can be read into OPDATA by issuing the command:

```
READ@BINARY FINAL.DAT NUM,X,XP,Y,YP,PHI,DE,HITS,XB,XPB,EX,EY,DQX,DQY (14R4)
```

From this data one can produce scatterplots of $\boldsymbol{x}$—$\boldsymbol{y}$, ε_x—ε_y, etc. In this case we plot $\boldsymbol{Q}_x$—$\boldsymbol{Q}_y$ by executing the following OPDATA procedure:

```
T=450                                !synchronous kinetic energy
E0=938.26                            !rest energy
GAMS=1+T/E0                          !synchronous gamma
DPP=(GAMS/(GAMS+1))*DE/T             !vector of dP/Ps
QX0=3.79                             !base tune X
QY0=5.71                             !base tune Y
CX=-4.018955                         !X chromaticity
CY=-6.827352                         !Y chromaticity
QX=QX0+DQX+CX*DPP                    !vector of X tunes
QY=QY0+DQY+CY*DPP                    !vector of Y tunes
PS 3.55 3.85 5.55 5.85               !plot scales
&NLXINC 6                            !tic marks
&NLYINC 6                            !tic marks
&NXGRID 1                            !grid spacing
&NYGRID 1                            !grid spacing
XLABEL "Q<d>X"                       !axis label
YLABEL "Q<d>Y"                       !axis label
PCHAR -2                             !plotting character
OPEN QXQY                            !open EDGR file for image
PLOT QX,QY                           !produce scatterplot
```

Using EDGR, the resulting image file can be edited or annotated and a publication-quality hardcopy produced, as shown in Figure 6.

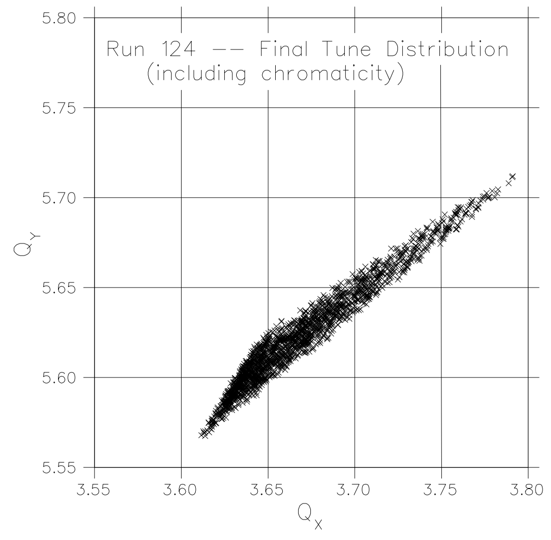

Figure 6: OPDATA/EDGR example.

Other possible uses of these utilities are for doing plots and FFT's of test particle tracking data, binning of particles for statistics, and plotting of time-varying quantities such as the painting functions and loss rates. A summary of the various output files produced by ACCSIM for post-processing can be found in Sections 30 and 31.

20 Program Design

The functional requirements of ACCSIM can be summarized by the following list of operations:

- Read main input file
- Read lattice file and construct the node list
- Perform preliminary calculations:
  - Compute constants used during the run
  - Initialize auxiliary routines
  - Initialize graphics devices
- Display node list and calculated parameters for the run
- Read and execute commands:
  - Execute turns

- Tabulate statistics
- Produce scatterplots
- Produce auxiliary output files
- Save and restore
- Other control and diagnostic functions

20.1 Computational core

The main loop invoked by the `TURN` command is the computational heart of the program and consists of the following sequence of steps:

1. **Begin loop on turns:**
2. Find current values of rf voltage and painting functions which depend on elapsed time.
3. Inject new macroparticles if required by continuous injection scheme.
   - Count injected particles that miss the foil and and tag them to prevent further tracking.
4. If there is an injection foil, check each particle in the ensemble for foil traversal. For each particle hitting the foil:
   - Increment hit counter
   - Increment foil traversal histogram
   - Find plural or multiple scattering angle and apply to particle
   - Find Landau energy loss and apply to particle
   - Increment foil energy loss histogram
5. Calculate ε_x and ε_y for each particle and tag particles exceeding acceptance.
6. **Begin loop on nodes:**
7. If plot flag is set for this node and this is a plotting turn:
   - Calculate line density and longitudinal space charge voltage
   - Produce scatterplot
8. If track flag is set for this node, write test particle data to track file.
9. If this node has an rf cavity, calculate and apply energy increment for each particle.
10. If this node is designated as a longitudinal space charge update node:
    - Calculate line density and space charge voltage.

- Apply energy increment to each particle based on distance to next space-charge update node.

11. If this node has a thin lens, calculate and apply angular kicks to particles.

12. If this node has an aperture, check particles for loss. For each lost particle:

    - Tag particle to prevent further tracking.
    - Increment loss counter.
    - Record turn number and node number.

13. If this node has a collimation slit, check for particles hitting the slit material. For each such particle:

    - Increment slit traversal counter.
    - Find multiple scattering angle and apply to particle.
    - Find Vavilov energy loss and apply to particle.

14. Advance particles to the next node by applying transfer matrix.

15. Increment the node counter.

16. **On arrival at the injection point, continue with Step 17. Otherwise go to Step 7 to process the next node.**

17. If chromaticity is enabled, calculate and apply chromatic tune shift for each particle.

18. If the transverse space charge calculation is enabled, the following is done on printing turns. If individual particle tune shifts are enabled, the following is done every turn.

    - Calculate line density.
    - Calculate emittances, figures of merit, and central tune shifts.
    - If enabled, calculate individual tune shifts and apply to particles.

19. If this is a printing turn, print a line of tabular output showing statistics, sweep values, and figures of merit.

20. Update the clock and turn counter to the end of the turn.

21. **If more turns, go to Step 2.**

20.2 Principal routines

While it is not intended to give an exhaustive description of the program structure, some of the important routines are outlined below:

Main Program Reads main input file, performs initialization, reads command stream, and executes turn loop.

READ_LATTICE Reads the lattice file (DIMAD output), extracts parameters and transfer matrices, and constructs the node list.

INJECT Generates and injects particles according to specified distributions.

RANSCAT Computes plural scattering angles using Keil distributions.

ASCAT Computes plural scattering angles using iterated single scatter.

LANDAU Calculates energy loss using Landau distribution.

VAVILOV Calculates energy loss using Vavilov distribution.

LINEDENS Bins particles longitudinally and calculates line density.

LSCHARGE Finds derivative of line density, calculates space charge voltage, and finds energy increments for particles.

THIN Performs thin-lens computations.

APER Performs aperture checks and simulates collimation slits.

LOSE Records particle losses.

CHROM Calculates chromatic tune shifts and applies them to particles.

DQ Contains supporting routines for transverse space charge.

EMITTANCE Bins particles in emittance space, activates transverse space charge calculation, applies tune shifts, and calculates figures of merit.

SCATPL Produces scatterplots of beam distributions.

STAT Compiles and prints statistics for the run.

EMIT Calculates and prints beam emittances.

LOSS Prints a list of lost particles.

21 Performance Issues

The principle application of ACCSIM has been to simulate multiturn injection into an accumulator ring for a Kaon Factory with the TRIUMF cyclotron as injector. This configuration dictates an injection cycle of approximately 20,000 turns. The large number of turns entails a considerable requirement in cpu time for a typical run in a VAX environment. Therefore ACCSIM has to be somewhat of a balancing act between meaningful results and acceptable turnaround times.

Some flexibility has been built into the program to allow the size and scope of runs to be tailored to the immediate requirements of the application. The main constraints to be dealt with are on the array capacities and on the calculations in the main loop. Increasing the array sizes yields higher macroparticle capacity and better statistics but may increase the virtual memory requirements to the point that frequent "paging" to disk will cripple the program. Carrying out tracking calculations to higher order may drive up cpu time requirements beyond reasonable limits when many thousands of turns must be simulated.

At present, the program capacity has been set at 10,000 macroparticles, however this limit can be easily changed to suit the local hardware, as described in Section 22. It may be desirable to increase this capacity in cases where a more powerful processor is available or where the application only requires a few hundred turns instead of several thousand.

Regarding the source code design, experience has shown that today's optimizing compilers do an excellent job of producing efficient machine code, and will do most of the things that programmers have traditionally done at the source level to squeeze more speed out of the code. One area where some speed can be gained is in array lookups: these can be minimized by storing repeatedly-used array values as scalars, and some care has been taken to do this in the main computational loops.

21.1 Run times

Production runs for the Kaon Factory accumulator ring have utilized the mid-range VaxStation-3200 workstation which, being a single-user or few-user system, has acquitted itself rather well in this application. This workstation's floating-point performance is rated at about 3 VAX-780 equivalents or about 0.5 VAX-8650. Timings for typical ACCSIM accumulation runs of 16,000 turns are:

Macroparticles	Foil	Space charge	Collimators	Cpu time (h:m)
2000	no	no	no	1:09
2000	yes	no	no	1:14
2000	yes	long.	no	1:35
2000	yes	long.	yes	2:39
2000	yes	long.+trans.	no	9:50
8000	yes	long.	no	6:04
8000	yes	long.+trans.	no	23:05

On a workstation used by one or two persons, the resulting turnaround times are a matter of a few hours for basic runs, overnight for runs with active transverse space charge, and a few days if all the stops are pulled out.

21.2 Benchmarks

A no-frills benchmark version of the code has been ported successfully and tested on a number of different architectures. The following table shows the results of these tests for a 20,000-turn accumulation run:

Machine	**Cpu time (m)**
TEKTRONIX 4320 (no fpa)	358
VAXSTATION 2000	339
VAXSTATION II/GPX	337
MICROVAX II	333
SUN 3/60	260
VAX 780	187
TEKTRONIX 4320 (fpa)	150
APOLLO DM3500 (no fpa)	148
VAXSTATION 3200	75
APOLLO DM3500 (fpa)	72
VAX 8600	50
VAX 8650	37
SCS40	26
DECSTATION 3100	23
APOLLO DN10000	17
DECSTATION 5000	15
CRAY XMP-22 (scalar mode)	3

The program has not yet been adapted to a vector- or parallel-computing environment, but some potential for vectorization should exist since the bulk of the computations are carried out in a series of loops over the particle ensemble arrays. These loops are independent units and are indexed by particle number, carrying out the same operation on each particle's coordinates. At least in principle, these operations can be carried out in parallel.

22 Modifying ACCSIM

Since the program has been decomposed into individual routine files, it is a relatively easy matter to make changes or add new routines. The basic procedure is:

```
$ FORTRAN ROUTINE1,ROUTINE2,...       !compile changed/new routines
$ LIBR/LOG ROUTINE1,ROUTINE2,...      !install then in the library
$ @ACCSIM.LNK                         !execute the link procedure
```

One straightforward modification that may be necessary is to change the array capacities. The program is currently set for a maximum of 10,000 particles and 100 nodes. These limits are set by parameter statements:

```
PARAMETER(NPMX=10000)
PARAMETER(NNMX=100)
```

The capacities can be changed by modifying these parameter statements in each source code file where they occur, then recompiling these source codes and relinking the program as described above.

Another obvious modification is to add a new node property. This can be done by declaring an integer array of size `NNMX` and adding it to the NODES input list in the main program. Example:

```
INTEGER IPROP(NNMX)
   ...
NAMELIST/NODES/ ...,IPROP
```

A nonzero value of `IPROP(I)` will indicate that node `I` has the property (for examples see Section 25). A module PROP similar to modules such as THIN or APER, should be written to implement the calculations simulating the node property. This module is then conditionally activated in the main computation loop:

```
IF (IPROP(INODE).NE.0) CALL PROP(INODE,IPROP(INODE),...)
```

In this example, the node number `INODE` and the value of `IPROP` at the node have been passed to the module where presumably they will be used in the calculation. Any other array values indexed by `INODE` that are needed by the module can also profitably be passed in this manner since it eliminates array look-ups.

Part II
Reference Guide

23 Main Input File

The primary input file is read from Unit 1. It consists of a title line and three parameter lists. The layout is as follows:

```
Title line
 $BEAM
  parameter=value, parameter=value, ...
        ...
 $END
 $NODES
  parameter=value, parameter=value, ...
        ...
 $END
 $MAIN
  parameter=value, parameter=value, ...
        ...
 $END
 $OPTIONS
  parameter=value, parameter=value, ...
        ...
 $END
```

The title line must be the first line of the file. The title will be reproduced on output files and plots to identify them. The parameter lists are identified as follows:

`$BEAM`	Injected beam and painting parameters
`$NODES`	List of nodes and node properties
`$MAIN`	Machine and control parameters
`$OPTIONS`	Special options parameters

Each list name *must begin in column 2* of the input file. Other than this restriction, the parameter lists are essentially in *free format.* No ordering of parameters is required and parameter assignments can be separated by blanks, commas or newlines. Examples of parameter assignments are:

1. Real parameters `A`:

   `A=5` or `A=5.0` or `A=0.5E1`

2. Real array **`B`** with 5 elements:

```
B=1,-2,4.2,79,6.2E2
```
or
```
B= 1 -2 4.2 79 6.2E2
```

3. Logical parameter **`C`**:

`C=TRUE` or **`C=T`**

Comments, indicated by a preceding "`!`" character, can be placed anywhere in a parameter list. If the whole line is a comment, the "`!`" must be in *column 2* or after.

The following sections describe the individual parameters for each list. Unless otherwise indicated, any required parameter that is omitted from the input will take on a default value of *zero* for numerical parameters and *false* for logical parameters. Some parameters are required only for certain program options and can otherwise be omitted. In the following sections, such parameters are tagged according to the program options they are relevant to.

Arrays are indicated in the lists by an array size in parentheses. For instance, the arrays describing the node properties (see Section 25) have size "(NNODE)" where NNODE is the parameter giving the number of nodes. Arrays can be entered as sequential lists of values, as shown above, or individual locations can be set, e.g.:

```
A(5)=2.7E-3  A(7)=4.0009
```

As with scalars, uninitialized array locations default to zero.

24 BEAM Parameter List

24.1 Transverse distributions

The horizontal and vertical injected beam phase-space distributions have the same parameter scheme. The following list shows the parameter names for the horizontal phase-plane. Replace X by Y to get the corresponding names for the vertical plane.

IDISTX	Injected beam distribution type: 0 No distribution: inject all particles at beam center (default) 1 Uniformly populated upright ellipse 2 Uniformly populated ellipse with Twiss parameters 3 Parameterized binomial distribution

IDISTX=1

DELTX	Half-width in x (mm) of upright ellipse
DELTXP	Half-width in x' (mr) of upright ellipse

IDISTX=2

ALPHX	Twiss parameter α of ellipse
BETX	Twiss parameter β of ellipse (m)
EPSX	Twiss parameter ε of ellipse (mm-mr). Area of ellipse is $\varepsilon\pi$.

IDISTX=3

AMX	Parameter m for binomial distribution. See Section 9.1 and Reference [6] for details. Examples are: $m = 0.0$ Hollow shell $m = 0.5$ Flat profile $m = 1.0$ Uniform (elliptical profile) $m = 1.5$ Elliptical (parabolic profile) $m = 2.0$ Parabolic $m \to \infty$ Gaussian
ALPHX	Twiss parameter α of ellipse
BETX	Twiss parameter β of ellipse (m)
EPSX	Twiss parameter ε for 2σ ellipse (mm-mr). Area of ellipse is $\varepsilon\pi$. The 2σ ellipse contains 86.5% to 100% of the beam depending on the distribution type (value of m). *Alternatively, specify* EPSLX.
EPSLX	Twiss parameter ε for limiting ellipse (mm-mr). Area of ellipse is $\varepsilon\pi$. The limiting ellipse contains 100% of the beam. *Alternatively, specify* EPSX.

24.2 Longitudinal distribution

IDISTL	Injected bunch longitudinal distribution type: 0 No distribution: inject all particles at beam center (default) 1 Uniformly populated rectangle 2 Uniformly populated ellipse 3 Parameterized binomial distribution

IDISTL=1,2

DELTE	Energy half-width of injected bunch (MeV)
DELTPHI	Phase half-width of injected bunch (deg)

IDISTL=3

AML	Parameter m for binomial distribution. For distribution types see parameter AMX above.
PHIM,DEM	Phase (deg) and energy (MeV) half-widths of 2σ ellipse for injected bunch. The 2σ ellipse contains 86.5% to 100% of the beam depending on the distribution type. *Alternatively specify* PHIL and DEL.
PHIL,DEL	Phase (deg) and energy (MeV) half-widths of limiting ellipse for injected bunch. The limiting ellipse contains 100% of the beam. *Alternatively specify* PHIM and DEM.

All distributions

PHISEP	Phase separation (deg) for double bunch. Default=0 (single bunch).

24.3 Dispersion matching

DMATCH	If TRUE, injected beam is dispersion-matched to machine. Default is FALSE.
ETAXI	[DMATCH=F] Injection-line x dispersion (m) at injection point. Default=0.
ETAXPI	[DMATCH=F] Injection-line x' dispersion (m) at injection point. Default=0.

24.4 Injected beam center

X0, XP0	Coordinates x and x' of center of injected beam (mm, mr)
Y0, YP0	Coordinates y and y' of center of injected beam (mm, mr)
PHI0	Central phase ϕ (deg) of injected bunch
DE0	Central energy displacement $\Delta E = E - E_s$ (MeV) of injected bunch, with reference to the synchronous energy E_s. *Alternatively, specify* DPP0.
DPP0	Central $\Delta p/p_s$ of injected bunch, with reference to the synchronous momentum p_s. *Alternatively, specify* DE0.

24.5 Distributions read from file

INJFILE	If TRUE, particle coordinates for the injected beam are read from a file (Unit 4) instead of being generated from Monte Carlo distributions. In this case the distribution parameters IDISTX, IDISTY and IDISTL, described above, are ignored. The file must contain a record for each particle to be injected. The record format is: $$\Delta x \quad \Delta x' \quad \Delta y \quad \Delta y' \quad \Delta\phi \quad \Delta(\Delta E)$$ where $\Delta x = x - x_0, \ldots, \Delta(\Delta E) = \Delta E - \Delta E_0$ are the deviations from the injected beam center $(x_0, x'_0, y_0, y'_0, \phi_0, \Delta E_0)$. The use of deviations rather than absolute quantities allows the beam to be painted.

24.6 Closed orbit bump

XCO	x coordinate (mm) of closed orbit at injection point. This is a fixed value and is used only if there is no sweep of the x closed orbit.
YCO	y coordinate (mm) of closed orbit at injection point. This is a fixed value and is used only if there is no sweep of the y closed orbit.

24.7 Painting

NXCO	Number of values for X closed orbit bump. Default=0 (closed orbit is fixed).
TXCO (NXCO)	[NXCO>0] List of time values (ms) for sweep.
VXCO (NXCO)	[NXCO>0] List of closed orbit values (mm), corresponding to the time values.

NYCO	Number of values for Y closed orbit bump. Default=0 (closed orbit is fixed).
TYCO (NYCO)	[NYCO>0] List of time values (ms) for sweep.
VYCO (NYCO)	[NYCO>0] List of closed orbit values (mm), corresponding to the time values.

NY0	Number of values for Y0 sweep. Default=0 (no sweep).
TY0 (NY0)	[NY0>0] List of time values (ms) for sweep.
VY0 (NY0)	[NY0>0] List of Y0 values (mm), corresponding to the time values.

NPHI0	Number of values for PHI0 sweep. Default=0 (no sweep).
TPHI0 (NPHI0)	[NPHI0>0] List of time values (ms) for sweep.
VPHI0 (NPHI0)	[NPHI0>0] List of PHI0 values (deg), corresponding to the time values.

NDE0	Number of values for DE0 sweep. Default=0 (no sweep).
TDE0 (NDE0)	[NDE0>0] List of time values (ms) for sweep.
VDE0 (NDE0)	[NDE0>0] List of energy offset values (MeV), corresponding to the time values.

25 NODES Parameter List

The node structure of the machine is described by the parameter NNODE (number of nodes) and a set of arrays of size NNODE which describe the the node properties. By default, the array entries are zero. Properties are enabled for a node by setting its array entries to nonzero values. Any number of properties can be set for a given node.

NNODE	Number of nodes, including the injection point (node 1). Thus NNODE must be at least 1. At present, the maximum number of nodes is 100.

NLABEL (NNODE)	An array of descriptive labels, one for each node. Each label is a string of up to 8 characters. By default, labels are blank. They are not interpreted by the program but are used to identify the nodes in program output.

ICAV (NNODE)	*RF cavity.* ICAV(I)=1 indicates an RF cavity at node I. The cavity will have voltage VRF/NCAV where VRF is the total RF voltage and NCAV is the number of cavities. VRF and NCAV are specified in the MAIN parameter list (see Section 26.2).

ILSC (NNODE)	*Longitudinal space charge update.* If ILSC(I)=1, the space charge potential will be calculated at node I and the space-charge energy gain to the next space-charge node will be applied. Usually nodes with ICAV=1 will also have ILSC=1. Additional updates between cavities will sometimes improve the accuracy of longitudinal tracking.

ITHIN (NNODE)	*Thin lens.* ITHIN(I)>0 indicates a thin lens at node I. Values for ITHIN are: 1 Dipole ($n = 0$) 2 Quadrupole ($n = 1$) 3 Sextupole ($n = 2$) 4 Octupole ($n = 3$) 9 General multipole (not yet implemented)
KNL (NNODE)	[ITHIN>0] Integrated strength K_nL (m^{-n}) of the element, where $K_n = (1/B\rho)(\partial^n B_y/\partial x^n)$.
ISKEW (NNODE)	[ITHIN>0] Skew flag. If ISKEW=1 the element is rotated by $\pi/(2n+2)$.

IAPER (NNODE)	*Aperture or collimator slit.* IAPER(I)>0 indicates an aperture or collimator slit at node I. The arrays XA1,XA2,YA1,YA2 are used to determine the aperture or slit dimensions. Particles exceeding an aperture will be tagged as lost and will be removed from the circulating ensemble. Particles exceeding slit dimensions will traverse the slit material and will undergo multiple scattering and energy loss. Values for IAPER are: 1 X aperture. Loss occurs if $x <$ XA1 or $x >$ XA2. 2 Y aperture. Loss occurs if $y <$ YA1 or $y >$ YA2. 3 Rectangular aperture. Loss occurs if $x <$ XA1 or $x >$ XA2 or $y <$ YA1 or $y >$ YA2. 4 Elliptical aperture. Loss occurs if $x^2/\mathrm{XA2}^2 + y^2/\mathrm{YA2}^2 > 1$. 11 Vertical slit. Traversal of material occurs if $x <$ XA1 or $x >$ XA2. 12 Horizontal slit. Traversal of material occurs if $y <$ YA1 or $y >$ YA2.
XA1, XA2 (NNODE)	[IAPER>0] X aperture or slit dimensions (mm) as described above.
YA1, YA2 (NNODE)	[IAPER>0] Y aperture or slit dimensions (mm) as described above.
TSLIT	[IAPER=11,12] Thickness of slit material (cm).
ZSLIT	[IAPER=11,12] Atomic number of slit material.
ASLIT	[IAPER=11,12] Atomic weight of slit material.
RHOSLIT	[IAPER=11,12] Density of slit material (g/cm^2).
VISLIT	[IAPER=11,12] Ionization potential of slit material (eV).

IPLOT (NNODE)	Plot flag. Setting IPLOT(I)=1 will cause scatterplots to be made at node I on designated plotting turns, as determined by NPLOT (in MAIN parameter list) or by the soon-to-be implemented `PLOT` command. By default, scatterplots are made at the injection point (node 1) if no other plot flags are set.
ITRAC (NNODE)	Track flag. Setting ITRAC(I)=1 will cause test particle coordinates at node I to be written to the track file. By default, tracking data is written at the injection point (node 1) only.

26 MAIN Parameter List

26.1 Vacuum chamber

XVAC	Horizontal semi-radius (mm) of vacuum chamber, used in transverse space charge calculation.
YVAC	Vertical semi-radius (mm) of vacuum chamber, used in transverse space charge calculation.

26.2 RF System

NHARM	Harmonic number
TSYNC	Synchronous kinetic energy (MeV)
NCAV	Number of rf cavities
VRF	Total rf voltage in volts. This is a fixed value and may be omitted if V_{rf} is being swept.

NVRF	Number of VRF steps. Default=0 (constant VRF).
VVRF (NVRF)	[NVRF>0] List of rf voltage values in volts.
TVRF (NVRF)	[NVRF>0] List of time values (ms) corresponding to voltage values.
TORF	[NVRF>0] Time offset (ms) for VRF function: RF time = Real time + TORF.

HRF1	Harmonic number for added VRF harmonic (constant in time). Default=0 (no added harmonic).
VRF1	[HRF1>0] Voltage (v) of added harmonic
PRF1	[HRF1>0] Phase (deg) of added harmonic

HRF2	Harmonic number for added VRF harmonic (constant in time). Default=0 (no added harmonic).
VRF2	[HRF2>0] Voltage (v) of added harmonic
PRF2	[HRF2>0] Phase (deg) of added harmonic

The total rf voltage is:

```
V = VRF*SIN(PHI) + VRF1*SIN(HRF1*(PHI+PRF1)) + VRF2*SIN(HRF2*(PHI+PRF2))
```

26.3 Foil or target

FOIL	If TRUE, an injection foil will be simulated. Default is FALSE.
TARG	If TRUE, an internal target is to be simulated. Default is FALSE.

FOIL=T or TARG=T

XFMIN	X coordinate at inside edge (mm)
XFMAX	X coordinate at outside edge (mm)
YFMIN	Y coordinate at bottom edge (mm)
YFMAX	Y coordinate at top edge (mm)
BINSIZE	Size (mm) of square histogram bins for foil or target surface area. The maximum size of the bin array is 100×100.
THICK	Foil or target thickness in μg/cm^2. If THICK=0, the foil or target will have no angular scattering or energy loss effects. Default=0.

THICK>0

ZFOIL	Atomic number of material
AFOIL	Atomic weight of material
RHOFOIL	Density of material in g/cm
VIFOIL	Ionization potential of material in eV
KEIL	If TRUE, a plural scattering distribution (tabulated by E. Keil *et al*) will be used to generate scattering angles, provided we are in the plural scattering domain $m \leq 20$, where m is the average number of single scatters per foil traversal. If FALSE, an accumulated single-scatter model is used to calculate plural scattering angles. Default is FALSE.
AMU	Factor μ for atomic radius, used to calculate cut-off angle $\theta_{\rm min}$ and total cross section based on cut-off Rutherford scattering law. Keil and others use 0.885, Jackson uses 1.4. Use 1.35 for best fit to M. Butler's tabulated distributions for carbon. For details see Section 11.5
SIGT	Total scattering cross-section in cm^2. This is used to determine average number of scatterings, but does not affect the single-scattering distribution. If SIGT is not specified, the cross-section is calculated using AMU above.

26.4 Macroparticle injection

NPINJ	Number of particles per injection. Default=0 (no programmed injection).
INJINT	[NPINJ>0] Turn interval for injection: 0 NPINJ particles injected at beginning of run 1 NPINJ particles injected per turn >1 Average of NPINJ particles injected every INJINT turns
NPMAX	Maximum number of particles in the machine. Further injection is disabled when this number is reached.

26.5 Line density

NBINS	Number of bins for rf phase histogram and line density analysis. The line density is used in longitudinal and transverse space charge calculations.
NSMOOTH	Number of iterations of digital filter smoothing for line density profile.

26.6 Space charge

LONGSC	If TRUE, longitudinal space charge calculations are done and energy kicks are applied to the particles. Default is FALSE.
DQVAR	If TRUE, transverse space charge calculations are done and tune shifts are applied to the particles on every turn. Default is FALSE.
DQPAR	If TRUE, transverse space charge calculations are done only on printing turns, to calculate figures of merit. Tune shifts are not applied. Default is FALSE.

LONGSC=TRUE or DQVAR=TRUE or DQPAR=TRUE

NMACRO	Number of macroparticles equivalent to PREAL real particles.
PREAL	Number of real particles equivalent to NMACRO macroparticles. The ratio PREAL/NMACRO is used to calculate the number of real particles in the machine.
NSCMIN	Minimum number of injected macroparticles before space charge calculations are performed. This applies to both longitudinal and transverse space charge. Default=1.

LONGSC=TRUE

RDIM	Ratio b/a of vacuum chamber radius to beam radius, used in longitudinal space charge calculation.
NLSC	Used only for *longitudinal-only* tracking [LPONLY=TRUE]. Number of integration steps per cavity. Default=1. If the number of cavities is small this should be set >1 to ensure stability in the finite-difference algorithm for motion under space charge.

DQVAR=TRUE or DQPAR=TRUE

EMFRAC	Fraction of beam to be used in determining the maximum emittance used in binning particles and calculating figures of merit. For example, if EMFRAC=0.98 then, in each plane, the extreme particles having emittances in the top 2% will not be included in the calculation and, in case DQVAR=TRUE, will not receive tune shifts. EMFRAC must be in the range 0.95 to 1.0. Default=1.0 (use all particles).

26.7 Chromaticity

CHROM1	If TRUE, chromaticity (linear tune shift) is simulated. Values of X and Y chromaticity are read from the DIMAD file. Default is FALSE.

26.8 Acceptance

The acceptance check is done once per turn. Particles whose emittance (Courant-Snyder invariant) exceeds the acceptance of the machine will be tagged and counted but will continue to be tracked.

ACCEPTX	Horizontal acceptance in units of π mm-mr.
ACCEPTY	Vertical acceptance in units of π mm-mr.

26.9 Histograms and plotting

NPLOT	Plotting increment. Scatterplots will automatically be produced whenever the turn number is a multiple of NPLOT. Default=0 (no automatic plotting).
BINX	Size in mm of X bins for scatterplot histograms
BINY	Size in mm of Y bins for scatterplot histograms
BINDE	Size in MeV of ΔE bins for scatterplot histograms

26.10 Emittance fractions

NEMIT	Number of emittance fractions to be tabulated. When the `STAT` command is issued and NEMIT>0, a list of emittance values will be printed giving the fraction of beam exceeding each emittance. Default=0 (no table).
DEMIT	[NEMIT>0] Emittance increment for table (mm-mr). Emittance values will be I*DEMIT for I=1 through I=NEMIT.

26.11 Random number generator

IRAN	Integer seed value for random number generator. This should be a large odd number. Default=1111111111.

27 OPTIONS Parameter List

27.1 Option switches

The following parameters control diagnostic and test output and can implement some "experimental" features of the program.

IPTRACK	Particle number for test particle tracking output. By default, the test particle is particle 1, which is always injected at the center of the incoming beam distribution.
NOSCAT	If TRUE, angular scattering in the foil will be disabled. Default is FALSE.
NOEL	If TRUE, energy loss in the foil will be disabled. Default is FALSE.
LPONLY	If TRUE, particle tracking will be done only in longitudinal plane. Default is FALSE. In this option, the transverse beam and the node structure are ignored and a simple machine model of NCAV equally-spaced rf cavities is used.
LSYMM	If TRUE, the longitudinal line density is forced to be symmetric about $\phi = 0$, by merging bin contents. Default is FALSE.
LCORR	If TRUE, a correction is made to keep the bunch centered at $\phi = 0$, each time the line density is calculated. Default is FALSE.
LMULTIP	If TRUE, longitudinal dipole and quadrupole amplitudes are calculated once per turn and written on Unit 24. Default is FALSE.
LDIAG	If TRUE, the space-charge potential at $\phi = 0$ is written once per turn on Unit 34. Default is FALSE.
SLLOG	If TRUE, collimator slit traversals are logged on Unit 34. Default is FALSE.
HLOG	If TRUE, foil or target traversals are logged on Unit 26. Default is FALSE.

27.2 Transverse space charge diagnostics

If DQVAR=TRUE or DQPAR=TRUE, the following diagnostic parameters can be set:

IMAGE	If set to 999, calculates image coefficients. Default=0.
IOPT	If set to 1, simulates a K-V beam. If set to 2, simulates a parabolic beam. Default=0.

IPRINT (6)	Array of flags for diagnostic output. Default=0.
	
	IPRINT(1)=1: Tune-shift diamond printed from PARAM (Units 2 & 6)
	IPRINT(2)=1: Additional data from PARAM (Units 2 & 6
	IPRINT(3)=1: Tune shifts from EMITTANCE (Unit 27)
	IPRINT(4)=1: G factor calculation from EMITTANCE (Unit 31)
	IPRINT(5)=1: Arrays from AVER (Unit 31)
	IPRINT(6)=1: Arrays from EXPR (Unit 31)

28 Interactive Commands

The interactive command stream is read from Unit 5, which is normally attached to the terminal in an interactive run and to the batch command file for a batch run. Commands are used to control the progress of the run and to invoke various functions that may need to be activated at arbitrary times during the run. In particular, commands typed at the terminal enable the program to be used in an experimental or diagnostic mode where particle coordinates can be displayed or altered, scatterplots can be produced, and program variables can be dumped to files.

Some commands have optional or required parameters. These are shown in the syntax descriptions. For example,

```
COOR  ipar1  [ipar2]
```

indicates that the `COOR` command requires a parameter `ipar1` and has an optional second parameter `ipar2`.

28.1 Tracking and analysis commands

TURN Syntax: `TURN nturn [nprint]`
Causes `nturn` turns to be executed. If `nprint` is specified, tabular output is printed every `nprint` turns. This output includes elapsed time, traversals, sweep values, emittances, and form factors. If `nprint` is not specified it is taken to be equal to `nturn`. If a continuous injection scheme is in effect, particles will be injected as the turns are executed.

INJ Syntax: `INJ npart`
Causes `npart` particles to be immediately injected into the machine. This is irrespective of any continuous injection scheme that may be in effect.

PAR A diagnostic command that prints on the screen a list of all input parameters and important calculated parameters for the run. This is the same list that is written to Unit 2 (interactive) or Unit 6 (batch) at the beginning of a run.

TIME Syntax: `TIME [tstep] [curinj]`
Calculates and prints the number of turns equivalent to the time step `tstep`, measured in milliseconds. If, in addition, the parameter `curinj` is specified, the number of real particles injected during the time step will be printed. Here, `curinj` is the injection current measured in microamps. These calculations are performed for reference only and do not affect other program operations. If the parameters `tstep` and `curinj` are omitted, the elapsed time and elapsed turns are printed.

STAT Compiles and prints various statistics for the current state of the run. These include counts of circulating and lost particles, emittances of beam fractions, foil traversal rate, and summary of losses.

COOR Syntax: `COOR ipar1 [ipar2]`
A diagnostic command that prints the coordinates of particles in the range **ipar1** through **ipar2**. Omit the second parameter to print only the coordinates of the single particle **ipar1**. Particles are numbered in the order in which they are injected into the machine. Particle 1 is designated as the reference particle and is always injected at the center of the incoming beam.

EMIT For each plane, estimates the beam emittance by fitting an ellipse to 99% and 95% of the particles. Also calculates the "4 RMS" emittance based on the ellipse containing two standard deviations of the beam in each dimension.

DUMP Writes coordinates for all particles on Unit 22. The record format is described in Section 31.

LINE Writes the current line density information on Unit 23. The record format is described in Section 31.

SET Syntax: `SET [ipar]`
SET is used to assign new coordinates to particle number **ipar**. The coordinates will be prompted for by the program. For example, the command **SET 1** can be used to set the coordinates of the reference particle for subsequent tracking with the **TRACK** command.

TRACK Initiates output of tracking data for the test particle. The test particle is chosen by input parameter IPTRACK (see Section 27) which defaults to particle 1. The data will be written at every node for which the ITRACK flag has been set (see Section 25), or by default at the end of every turn. The record format is described in Section 31.

LOSS Produces a list of particle losses, giving the turn numbers and nodes where the losses occurred, and the coordinates x, y, and ΔE of particles at the time of loss.

TEST Provides a testing facility for the scattering angle generator. A specified number of traversal events will be simulated and the scattering angles written to Unit 32 (for Keil or Molière treatment, Sections 11.3 and 11.2) and Unit 33 (for repeated-single-scattering model, Section 11.4).

FHIST Writes the 2-dimensional foil histograms on Unit 41. One record is written for each bin (i, j). The format is i, j, N, E where N is the number of traversals accumulated in the bin and E is the total energy in MeV lost during those traversals (see Section 11.8).

FOIL Calculates projected heat distributions (horizontal and vertical) in the foil, based on energy lost by particles during foil traversals. Distributions are normalized to a peak value of 1. For a given circulating current, entered by the user, the rate of foil heating (average foil power in watts) will be calculated and printed. This data is written to the screen and to Unit 42 for further processing.

SAVE Syntax: `SAVE filename`
This command saves the state of the run, including internal variables, arrays and time-dependent parameters, on Unit 11 for later restoration by the `RESTORE` command. Note that the input parameters from the main input file are not saved. Therefore, to get an exact restoration, the restoration run must use the same input file. This allows the possibility of resuming a run with changed input parameters.

RESTORE Syntax: `REST filename`
This command restores the state of a run by reading on Unit 12 a file created by the `SAVE` command. The run can then be continued from the point at which the save file was made. As noted above, the save file does not contain the main input file parameters, which will remain at their values as read from the input file for the current run.

HELP Displays on the screen a summary of all the interactive commands.

TERMINAL This command is used only in interactive mode. If the command input stream (Unit 5) is currently being read from a file, this command closes that file and attaches the input stream to the terminal so additional commands can be typed interactively.

STOP or CTRL/Z Terminates the command input stream and exits from the program. In the absence of a `STOP` command, the program terminates whenever an end-of-file is encountered on the command input stream (Unit 5).

28.2 Graphics commands

SCAT Produces scatterplots and histograms of particle positions in the planes X_b—X_b', Y—Y', X—Y, and ϕ—ΔE. X_b and X_b' are the horizontal phase space coordinates for each particle:

$$\begin{aligned} X_b &= X - \eta_x \frac{\Delta p}{p} - X_{co} \\ X_b' &= X' - \eta_x' \frac{\Delta p}{p} \end{aligned}$$

where X_{co} is the closed orbit bump. The smoothed line density of the bunch is shown above the longitudinal plot. If longitudinal space charge is in effect (LONGSC=TRUE), the space charge voltage per turn is also plotted as a function of rf phase, and the effect of this potential will be shown on the separatrix of the longitudinal plot.

SCL Produces plots only for the longitudinal plane: scatterplot, histograms, line density and space charge voltage.

SLICE Produces scatterplots of "phase slices", as a means of visualizing the physical shape of the beam. Particles are binned in rf phase and an X–Y scatterplot is produced for each phase bin. The scatterplots are combined in a single picture where they are referenced to a phase axis.

EPS This command produces a scatterplot of $\sqrt{|\boldsymbol{x}|\boldsymbol{\beta}_x}$ versus $\sqrt{|\boldsymbol{y}|\boldsymbol{\beta}_y}$.

HARD In interactive mode only, this command produces an immediate hard copy of the last plot that was made, by queueing the plot file for printing on the queue pointed to by the logical name HP$LASER (for HP Laserjet) or LN03$LASER (for LN03+). This command is invalid for GKS metafiles.

GPLOT This command brings up the GPLOT parameters file (Unit 8) for editing. This file contains parameters determining the scaling and other characteristics of the plots produced for the run. After editing the parameters as desired, the current plot, if any, can be redrawn with the new characteristics by re-issuing the relevant plotting command. The graph parameters are discussed in Section 29 and are documented in detail by the TRIUMF Manual *Graph Plotting and Low Level Graphics*[33].

CL Clears the graphics and alphanumeric screens of the terminal.

HIRES Increases the resolution of HP Laserjet graphics output from 100 dots per inch to 150 dots per inch. Does not affect LN03+ or GKS output.

PIX Sets the size in pixels of the dots used to represent macroparticles in the scatterplots. The default is 1 pixel.

DWG Opens a device-independent drawing file to receive the subsequent graphics output. This file can be read and edited by the EDGR[34] graphics editor, and output can be produced on many different hardcopy devices.

29 Plotting Parameters

The plotting parameters file controls the layout and scaling of the various graphs produced by ACCSIM. This file is read on Unit 8 whenever a plot is produced. It consists of a series of records assigning values to parameters. Example:

```
'NXDEC' -1
'NYDEC' -1
'XNUMSZ' 8
'XLABSZ' 8
'YNUMSZ' 8
'YLABSZ' 8
'%XLAXIS' 15
'%XUAXIS' 85
'%YLAXIS' 15
'%YUAXIS' 77
'XTICA' 90
'YTICA' -90
'%XITICL' 1.5
'%YITICL' 1.5
'XAUTO' 0.
'YAUTO' 0.
'XPAUTO' 0.
'YPAUTO' 0.

'PLOT#' 1     /Xb-X'b
'XLABEL' 0
'Xb (mm)'
'YLABEL' 0
'X''b (mr)'
'XUWIND' 320
'YLWIND' 280
'XMIN' -60
'XMAX' 60
'NLXINC' 6
'NSXINC' 2
'YMIN'-8
'YMAX' 8
'NLYINC' 4
'NSYINC' 1

'PLOT#' 2       /Y-Y'
'XLABEL' 0
'Y (mm)'
'YLABEL' 0
'Y'' (mr)'
'XLWIND' 320
'YLWIND' 280
  ... etc.
```

The first section of the file consists of *global* parameter assignments that will apply by default to all plots. This is followed by a series of sections for each type of plot to be used. These sections assign specific parameters and can also override the global parameters. The start of each section is identified by a unique plot number assigned to the **PLOT#** parameter. The plot types are:

Section	**Plot type**	Command
1	X_b—X_b (X phase space)	**SCAT**
2	Y—Y'	**SCAT**
4	X—Y	**SCAT**
3	ϕ—ΔE	**SCAT**
8	V_{sc} (space charge potential)	**SCAT**
9	ϕ—ΔE	**SCL**
10	V_{sc} (space charge potential)	**SCL**
6	$\sqrt{\lvert x\rvert/\beta_x}$—$\sqrt{\lvert y\rvert/\beta_y}$	**EPS**
7	X—Y phase slice	**SLIC**

The sections can appear in any order in the file. Entries are required only for the plot types actually used during a run.

29.1 GPLOT parameters

The different named parameters control many different aspects of the plot. All parameters are described in detail in Reference [33]. For reference, some of the commonly-used ones are listed below. In the following, "X" can be replaced by "Y" to get the equivalent parameter name for the Y axis.

NXDEC	Number of decimal places for axis numbers point)
XNUMSZ	Size of axis numbers
XLABSZ	Size of axis label
%XLAXIS	Start position of axis (percentage of screen)
%XUAXIS	End position of axis (percentage of screen)
XTICA	Angle of tic marks, with respect to X axis
%XITICL	Axis number offset (percentage of screen)
XAUTO	Set to 0 to disable automatic scaling
XPAUTO	Set to 0 to disable automatic power
XLABEL	Axis label (placed on following line)
XMIN	Minimum value on axis
XMAX	Maximum value on axis
NLXINC	Number of large increments on axis
NSXINC	Number of small increments on axis
XLWIND	Lower X coordinate of output window
XUWIND	Upper X coordinate of output window

During an interactive ACCSIM run, the plotting parameters can be altered while plots are being viewed on the screen. To do this, issue the command (say **SCAT**) to produce the plot, and then issue the command **GPLOT**. This brings up the plotting parameters file for editing under the EDT editor. After entering new values for any parameters and exiting from the editor, the plot command can be re-issued, and the plot will be produced again using the latest version of the file.

30 I/O Streams

The following summarizes the various input and output streams by unit number. The record formats for the output streams are described in Section 31 following.

1	Main input file: injected beam and machine parameter lists.
2	Echo output file. This is a copy of the terminal stream (Unit 6) and provides an output file for interactive runs.
3	Lattice file. This is the DIMAD output file for the lattice.
4	Distribution input: used when injected beam distribution is read from a file.
5	Command input stream. This is the terminal in interactive mode and the batch command file in batch mode.
6	Main output stream. This goes to the terminal in interactive mode and the log file in batch mode. This stream is also sent to Unit 2 to produce an output file for interactive runs.
7	Plot file output stream. In batch mode, each plot image or GKS metafile frame is accumulated here for later printing. In interactive mode, GKS frames accumulate as in batch mode, while for the other devices a new file is opened for each image and the previous file is discarded. This allows an immediate hardcopy of any plot, using the `HARD` command.
8	GPLOT parameters file. This is used only if plots are made.
9	Injection coordinates file. The coordinates of each particle are recorded here when the particle is injected.
10	Tabular record file. The tabular output of statistics that is sent to Unit 6 during the run is also sent without annotation to this stream, for reading by external programs.
11	Save file output: used by `SAVE` command.
12	Save file input: used by `RESTORE` command.
21	Test particle tracking data. (`TRACK` command)
22	Particle coordinate dumps. (`DUMP` command)
23	Line density profile output. (`LINE` command)
24	Record of longitudinal dipole and quadrupole amplitudes.
25	Record of beam emittances.
26	Record of foil traversals.
31	Diagnostic output for DQ modules.
32	Keil or Moliere scattering angle test output. (`TEST` command)
33	PSCAT scattering angle test output. (`TEST` command)
34	Record of collimator slit traversals.
41	Foil energy deposition histogram output. (`FOIL` command)
42	Foil heat distribution output. (`FOIL` command)

30.1 File naming convention

Since all files are connected by logical assignments, any file names whatever can be used. In practice, it is useful to have a convention whereby files for a given run all have the same name, e.g. a run name or number, but have different 3-character file types to distinguish them. The following table shows such a set of file types:

Unit	*File*	*File type*
1	Main input	`.DAT`
2	Echo output	`.OUT`
3	Lattice file	`.LAT`
4	Distribution input	`.DIS`
5	Command input stream (batch)	`.COM`
6	Main output stream (batch)	`.LOG`
7	Plot file	`.PLT`
8	Plotting parameters	`.GPL`
9	Injection coordinates	`.INJ`
10	Tabular record	`.REC`
11	Save file output	`.SAV`
12	Save file input	`.SAV`
21	Test particle track	`.TRK`
22	Coordinate dump	`.DMP`

31 Output Formats

Unit 9: Injection coordinates

1	2	3	4	5	6	7	8
Par#	X	X'	Y	Y'	ϕ	ΔE	Turn

Unit 10: Tabular record file

1	2	3	4	5	6	7	8	9	10
Turn	Time	N_m	Hits	Avg Hits	Lost	X_{co}	Y_{co}	T_0	ΔE_0

11	12	13	14	15	16	17	18
T_{sync}	N_p	E_x	E_y	B_f	G_f	ΔQ_x	ΔQ_y

Unit 21: Test particle tracking

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Turn	X	X'	Y	Y'	ϕ	ΔE	Hits	X_b	X'_b	X_{co}	Y_{co}	ε_x	ε_y	ΔQ_x	ΔQ_y

Unit 21: Reference particle tracking – longitudinal

1	2	3	4
Turn	ϕ	ΔE	V_{sc}

Unit 22: Particle coordinate dump

1	2	3	4	5	6	7	8	9	10	11	12	13	14
Par#	X	X'	Y	Y'	ϕ	ΔE	Hits	X_b	X'_b	ε_x	ε_y	ΔQ_x	ΔQ_y

Unit 23: Line density profile output

1	2	3	4	5
ϕ (deg)	Time (μs)	Count	Smoothed count	V_{sc} (kv)

Unit 24: Longitudinal dipole and quadrupole amplitudes

1	2	3
Turn	Dipole	Quadrupole

Unit 25: Emittances

1	2	3	4	5	6	7
Turn	Hits	ε_x(min)	ε_x(max)	ε_x^f(min)	ε_x^f(max)	$\varepsilon_x(2\sigma)$

8	9	10	11	12
ε_y(min)	ε_y(max)	ε_y^f(min)	ε_y^f(max)	$\varepsilon_y(2\sigma)$

Unit 26: Foil traversals

1	2
ϕ	ΔE

Unit 26: Collimator slit traversals

1	2	3
Par#	Turn	Node

1	2	3	4	5	6	7	8
X	X'	Y	Y'	ΔE	dX'	dY'	dE

References

[1] F.W. Jones, G.H. Mackenzie and H. Schönauer, "ACCSIM – A Program to Simulate the Accumulation of Intense Proton Beams," *Proc. 14th International Conference on High Energy Accelerators*, in *Particle Accelerators* **31:**199 (1990).

[2] H. Schönauer, "Addition of transverse space charge to ACCSIM code," TRIUMF Design Note TRI-DN-89-K50 (1989).

[3] R. Baartman and F.W. Jones, "Transport equations for ACCSIM," TRIUMF Design Note TRI-DN-89-K62 (1989).

[4] R.V. Servranckx, K.L Brown, L. Schachinger, and D. Douglas, *User's Guide to the Program DIMAD*, SLAC Report 285 UC-28 (A), May 1985.

[5] K.L. Brown, D.C. Carey, Ch. Iselin and F. Rothacker, *TRANSPORT, A Computer Program for Designing Charged Particle Beam Transport Systems*, CERN 80-04, 1980.

[6] W. Joho, *Representation of Beam Ellipses for Transport Calculations*, SIN-REPORT TM-11-14, 8.5.1980.

[7] J.D. Jackson, *Classical Electrodynamics*, John Wiley & Sons, 1962.

[8] A.K. Kaminsky, R.A. Meshcherov and M.I. Popova, "The efficiency of charge-exchange injection in proton accelerators and storage rings," *Nucl. Instrum. Methods* **137:**183 (1976).

[9] E. Keil, E. Zeitler and W. Zinn, "Zur Einfach- und Mehrfachstreuung geladener Teilchen," *Z. Naturforschg.* **15 a**, 1031 (1960).

[10] G. Jarlskog and L. Jönsson, *Monte Carlo Generation of Multiple Scattering of Particles (MLR)*, CERN Program Library, W601 (1973).

[11] B. Schorr, *Programs for the Landau and the Vavilov Distributions and the Corresponding Random Numbers*, CERN Program Library, G110/G111 (1974).

[12] A. Thiessen, "PSCAT random number generator for plural scattering in stripper foil," LAMPF II Technical Note 85-007, LA-UR-87-1252.

[13] J.E. Freund, *Mathematical Statistics*, 2nd ed., Prentice-Hall, 1971.

[14] M.N. Butler, S. Koscielniak, G.H. Mackenzie and F. Jones, "Coulomb scattering cross sections for accelerator design," *Proc. 1989 IEEE Particle Accelerator Conference*, p. 654.

[15] M.N. Butler and S. Koscielniak, "Scattering cross sections for ^{9}Be, ^{12}C, ^{16}O and ^{27}Al," TRIUMF Design Note TRI-DN-89-K27 (1989).

[16] D. Raparia, *Charge Exchange Injection into an Accumulator Ring*, M.Sc. Thesis, University of Manitoba.

[17] H.G. Hereward, "What are the equations for the phase oscillations in a synchrotron?" CERN 66-6 (1966).

[18] S.R. Koscielniak, "The LONG1D Simulation Code," *Proc. 1st European Particle Acclerator Conference*, Rome, 1988, p. 743.

[19] S.R. Koscielniak, *LONG1D User Guide*, TRIUMF Design Note TRI-DN-88-20 (1988).

[20] A. Hofmann, "Single-beam collective phenomena — longitudinal," *Theoretical aspects of the behaviour of beams in accelerators and storage rings*, CERN 77-13 (1977).

[21] M.K. Craddock, *High Intensity Circular Proton Accelerators*, TRIUMF Report TRI-87-2 (1987).

[22] J. Wei, S.Y. Lee and A.G. Ruggiero, "The effect of longitudinal space charge on multi-turn capture in the AGS booster," *Particle Accelerators*, Vol. 23 (1989).

[23] A. Thiessen and J.L. Warren, "ARCHSIM: A proton synchrotron tracking program including longitudinal space charge," Los Alamos Preprint LA-UR-83-2703 (1983).

[24] F.B. Hildebrand, *Introduction to Numerical Analysis*, McGraw-Hill, 1956.

[25] F. Scheid, *Numerical Analysis*, McGraw-Hill, 1968.

[26] C. Bovet *et al.*, *A Selection of Formulae and Data Useful for the Design of A.G. Synchrotrons*, CERN/MPS-SI/Int. DL/70/4 (1970).

[27] M.R. Masullo and H. Schönauer, *Progress in Painting (Transverse Painting with Linear Coupling)*, EHF-88-7 (1988).

[28] K.H. Reich, K. Schindl and H. Schönauer, "An approach to the design of space-charge limited high intensity synchrotrons," *Proc. 12th International Conference on High-Energy Accelerators*, Fermilab, 1983, p. 438.

[29] R. Baartman, "The betatron phase shift approximation for simulations with transverse space charge," TRIUMF Design Note TRI-DN-90-K133 (1990).

[30] R. Baartman, "Symplectifying the case where transverse tune is a function of rf phase," TRIUMF Design Note TRI-DN-90-K120 (1990).

[31] P. Bennett and C.J. Kost, *OPDATA Reference Manual*, TRIUMF Computing Document TRI-CD-87-04 (1989).

[32] J.L. Chuma, *PLOTDATA User's Guide and Command Reference Manual*, TRIUMF Computing Documents TRI-CD-87-03a/b (1989,1990).

[33] J.L. Chuma, *Graph Plotting and Low Level Graphics*, TRIUMF Computing Document TRI-CD-87-02 (1989).

[34] F.W. Jones, *EDGR: Graphics Editor User's Guide*, TRIUMF Computing Document TRI-CD-87-01 (1990).

Appendix: Example Run

The attached pages contain example input and output for a typical run for TRIUMF's proposed Kaon Factory Accumulator Ring. This run simulates a full accumulation cycle of 16,000 turns or elapsed time of ~16 ms. A total of 8000 macroparticles are injected, simulating the accumulation of 1.39×10^{13} protons. The painting scheme comprises lowering of the vertical closed orbit from ~8 mm to 0 mm and ramping of the injection energy from 450.7 MeV to 451.9 MeV. Due to the achromatic injection and the ring dispersion of $\eta = 3.65$m, a horizontal painting effect occurs in conjunction with the energy ramping.

A 250 μg/cm^2 injection foil is used. The machine model has 4 nodes: the injection foil and three cavities with total voltage 514 kV and harmonic 45. The ring operates at 450 MeV with $\phi_s = 0$. Longitudinal space charge is simulated, with the energy increments being applied at the cavity nodes. The space-charge potential for the final ensemble is ~80 kV per turn. The "passive" transverse space charge calculation is enabled, that is, tune shifts are calculated but the particle coordinates are not revised. The final central tune shifts are $\Delta \boldsymbol{Q}_x = -0.156$ and $\Delta \boldsymbol{Q}_y = -0.121$.

Particles traverse the foil an average of 64 times, so emittance growth due to scattering is minimized. Particles scattered into large angles can be seen on the scatterplots. These comprise about 0.5% of the beam and would be removed by the collimation system, although this was not simulated in the run.

In the example the following files are reproduced or excerpted:

1. **`A121.COM`**, command file for the run.
2. **`A121.DAT`**, main input file.
3. **`A3.DMO`**, lattice file (DIMAD output).
4. **`A.GPL`**, plotting parameters file.
5. **`A121.LOG`**, main output file (batch log).
6. **`A121.PLT`**, scatterplots of accumulated beam.